1

ARI Research Note 89-04

Design of a MANPRINT Tool for Predicting Personnel and Training Characteristics Implied by System Design

Alfred O. Dick, Alvah C. Bittner, Jr.,
and Regina Harris
Analytics, Inc.
with Appendixes by Robert J. Wherry, Jr.

for

Contracting Officer's Representative
Christine R. Hartel

Manned Systems Group
John F. Hayes, Chief

Systems Research Laboratory
Robin L. Keesee, Director

January 1989

ARMY RESEARCH INSTITUTE

United States Army
Research Institute for the Behavioral and Social Sciences

UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a. REPORT SECURITY CLASSIFICATION
Unclassified

1b. RESTRICTIVE MARKINGS
--

2a. SECURITY CLASSIFICATION AUTHORITY
--

2b. DECLASSIFICATION / DOWNGRADING SCHEDULE
--

3. DISTRIBUTION / AVAILABILITY OF REPORT
Approved for public release;
distribution unlimited.

4. PERFORMING ORGANIZATION REPORT NUMBER(S)
--

5. MONITORING ORGANIZATION REPORT NUMBER(S)
ARI Research Note 89-04

6a. NAME OF PERFORMING ORGANIZATION
Analytics, Inc.

6b. OFFICE SYMBOL *(If applicable)*
--

7a. NAME OF MONITORING ORGANIZATION
U.S. Army Research Institute for the
Behavioral and Social Sciences

6c. ADDRESS *(City, State, and ZIP Code)*
2500 Maryland Way
Willow Grove, PA 19090

7b. ADDRESS *(City, State, and ZIP Code)*
5001 Eisenhower Avenue
Alexandria, VA 22333-5600

8a. NAME OF FUNDING / SPONSORING ORGANIZATION
Same as 7a

8b. OFFICE SYMBOL *(If applicable)*

9. PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
MDA903-86-C-0413

8c. ADDRESS *(City, State, and ZIP Code)*
Same as 7b

10. SOURCE OF FUNDING NUMBERS

PROGRAM ELEMENT NO.	PROJECT NO.	TASK NO.	WORK UNIT ACCESSION NO.
6.27.85	A791	1.2.1	1.2.1.C.2

11. TITLE *(Include Security Classification)*
Design of a MANPRINT Tool for Predicting Personnel and Training Characteristics Implied by System Design

12. PERSONAL AUTHOR(S)
Dick, Alfred O., Bittner, Alvah C., Jr., and Harris, Regina (R. J. Wherry, Jr., Appendixes)

13a. TYPE OF REPORT
Final

13b. TIME COVERED
FROM 87/06 TO 88/01

14. DATE OF REPORT *(Year, Month, Day)*
1989, January

15. PAGE COUNT
314

16. SUPPLEMENTARY NOTATION
Intelligent
Christine R. Hartel, Contracting Officer's ~~Representative~~
Armed Services Vocabulary Aptitude Battery

17. COSATI CODES

FIELD	GROUP	SUB-GROUP

18. SUBJECT TERMS *(Continue on reverse if necessary and identify by block number)*
MANPRINT, Performance, Design evaluation,
ASVAB, Modeling, Personnel, Aptitude tests.
~~HOS~~ Simulation, ~~Training~~

19. ABSTRACT *(Continue on reverse if necessary and identify by block number)*

This report describes the development and design specifications for a software-based aid for Army system designers. The purpose of the aid is to evaluate a system design by determining the operator and maintainer characteristics required by that design to reach criterion system performance levels. The aid is an analytical, simulation-based approach that will predict the impact on total system performance of individual differences in cognitive and psychomotor performance and of new system technologies. The specification describes a system based on the Human Operator Simulator (HOS) IV. In this approach, HOS performance micromodels are to be built from the Army Research Institute's (ARI) Project A predictor data. Although the design specification described herein is not being developed any further by ARI, it may prove useful for other projects.

20. DISTRIBUTION / AVAILABILITY OF ABSTRACT
☒ UNCLASSIFIED/UNLIMITED ☐ SAME AS RPT. ☐ DTIC USERS

21. ABSTRACT SECURITY CLASSIFICATION
Unclassified

22a. NAME OF RESPONSIBLE INDIVIDUAL
Jonathan Kaplan

22b. TELEPHONE *(Include Area Code)*
(202) 274-8873

22c. OFFICE SYMBOL
PERI-SM

Previous editions are obsolete.

SECURITY CLASSIFICATION OF THIS PAGE
UNCLASSIFIED

U.S. ARMY RESEARCH INSTITUTE
FOR THE BEHAVIORAL AND SOCIAL SCIENCES

A Field Operating Agency Under the Jurisdiction of the Deputy Chief of Staff for Personnel

EDGAR M. JOHNSON Technical Director	JON W. BLADES COL, IN Commanding

Research accomplished under contract
for the Department of the Army

Analytics, Inc.

Technical review by

J. Dexter Fletcher
Jonathan Kaplan

NOTICES

DISTRIBUTION: This report has been cleared for release to the Defense Technical Information Center (DTIC) to comply with regulatory requirements. It has been given no primary distribution other than to DTIC and will be available only through DTIC or the National Technical Informational Service (NTIS).

FINAL DISPOSITION: This report may be destroyed when it is no longer needed. Please do not return it to the U.S. Army Research Institute for the Behavioral and Social Sciences.

NOTE : The views, opinions, and findings in this report are those of the author(s) and should not to be construed as an official Department of the Army position, policy, or decision, unless so designated by other authorized documents.

DESIGN OF A MANPRINT TOOL FOR PREDICTING PERSONNEL AND TRAINING CHARACTERISTICS IMPLIED BY SYSTEM DESIGN

CONTENTS

LIST OF TABLES

Page

LIST OF TABLES (Continued)

LIST OF FIGURES

Page

LIST OF FIGURES (Continued)

Page

LIST OF FIGURES (Continued)

Page

LIST OF FIGURES (Continued)

DESIGN OF A MANPRINT TOOL FOR PREDICTING PERSONNEL AND TRAINING CHARACTERISTICS IMPLIED BY SYSTEM DESIGN

1. INTRODUCTION

This report presents a design specification for a tool for evaluating system designs in terms of the personnel characteristics required to ensure successful system operation -- System Evaluation for Personnel and Training Analysis (SEPTA). This tool is a candidate for one of the six products (specifically Product 6) being procured by the Army Research Institute for the Behavioral and Social Sciences (ARI) to support the Army's MANPRINT Methods Program. The objective of Product 6 is to predict the required characteristics of maintenance and operational personnel based on system design and mission performance requirements stated in terms of time and accuracy. Task 1 called for preparation of a detailed concept paper describing the candidate product; Task 2 calls for detailed design specification of the product; and Task 3 calls for implementation of the product. The material reported here represents accomplishment of Task 2 of the overall product development effort.

This report deviates from a traditional and true design specification in several ways. A design specification normally contains program descriptions and detail which should be sufficient for a programmer to implement the design in software. The present design incorporates state-of-the-art concepts in human modeling and simulation, human performance, rather sophisticated statistical concepts, the most current thinking in screen and user interface design as well as more traditional data processing concepts such as data base design and report generation. It would be unrealistic to expect a reader to be familiar with all of these topics and integrate them easily. Accordingly, and in line with the old expression, "There is nothing so practical as a good theory." we have taken the liberty of including more theory than would normally be expected. Further this type of discussion will be scattered throughout the report as introduction and support for the various modules.

The design that is described here for Product 6 is based on the use of several mathematical tools for efficient sampling of performance and extrapolation of profiles for performance and personnel characteristics. These are supported by powerful simulation tools; a) for prediction of key aspects of dynamic performance to determine performance speed and accuracy to compare with system requirements and; b) for evaluation of the degree to which personnel characteristics such as body size would enable specific individuals to perform at all with a given system in terms of physical accessibility of required performance conditions (i.e., reaching and seeing all necessary device components). The development of our Product 6 design has been explicitly opportunistic, seeking to adapt existing modeling and analysis tools to perform required functions within our product and to offer a similar opportunity to the eventual product users by enabling them to incorporate simulation components easily from outside sources into their analysis efforts. Overall usability of the product has been a key focus of the design effort.

This report documents the results of our efforts to create a design specification for Product 6. In order to establish the context for product development, the objectives of this effort with respect to the MANPRINT Methods Program are reviewed in Section 2; this includes proposed interconnections among Products 1 to 6 as well as the general context of application and some limitations. An overview of our overall design for Product 6 is presented in Section 3. Discussions of the components of Product 6 which will simulate human performance are subsequently offered in Section 4 through 9. Specifically, in Section 4 we discuss the user interface and in Section 5, the setup procedures required. In Section 6 we describe the dynamic simulation and in Section 7 the static simulation. Section 8 contains description of programs for summarizing and analysis of the results. Section 9 contains general components and subroutines which are used in a number of places.

Since the contract SOW for this effort calls for explicit attention to seven general issues (SOW Section 6.a), it may be useful to indicate where each of these issues is addressed in this report. These are as follows:

(1) *The required inputs of each product, its location, and how it will be accessed.*

(2) *The components of each product, where they will be obtained, and how they will interact.*

(3) *The processes by which the product will produce its outputs.*

(4) *The outputs of each product, described in exact man-machine interface terms.*

These are found in Sections 4 through 9. Specifically, the data sources required as well as the results of each module are discussed as part of the module.

(5) *The means by which computer files will achieve security.*

Some system software specifications were provided by ARI along with hardware specifications. These will be found in Appendix A.

(6) *The means to insure organizational acceptance of the outputs of each product.*

The first step to insure organizational acceptance is to provide usable results. This has been achieved through the design. The second step is to ensure that the results are in a form readily understood. This second step can only be accomplished completely by discussions with potential users.

(7) *The detailed schedule and estimate of costs for product development in 1986 dollars.*

The detailed schedule and estimate of costs are contained on separate sheets.

2. CONTEXT FOR SEPTA DEVELOPMENT

Understanding of the complex concept that is presented in this report for System Evaluation for Personnel and Training Analysis (SEPTA) requires recognition of the context of objectives and constraints that have guided its development. In this section, we discuss some organizational issues which guided and influenced the initial concept development. These issues are an important influence in determining the forms software must take, both from the user perspective and the organizational perspective. To start, we examine the technological context provided by the plan for the MANPRINT Methods Program and the designs for the other five MANPRINT Method Products (MMPs). We will also consider the major information processing objectives for SEPTA and some of their implications. Next we address the organizational context in which SEPTA will be used and the types of anticipated users. We will conclude with a discussion of some explicit limitations in the scope of the current effort.

2.1 MANPRINT

The Army Manpower and Personnel Integration (MANPRINT) initiative focuses on soldier-in-the-loop and front-end acquisition process. MANPRINT seeks to integrate six areas that are concerned with the soldier into the material acquisition process so that soldier needs and abilities are considered early in the process. The six domains are:

- Manpower,
- Personnel,
- Training,
- Human engineering,
- Health Hazards Assessment, and
- System Safety.

Manpower, Personnel, and Training (M^PT)

Manpower, personnel, and training are critical areas in the Army Materiel Acquisition Process. Manpower is concerned with force structure and deals with how many people and of what Military Occupational Speciality (MOS) are needed to operate, maintain, and support materiel. Personnel issues deal with **the kind of people** needed to maintain, operate and support materiel. Training, of course, is the **instruction of the soldier** in specific skills and procedures needed to perform necessary tasks.

Human Factors Engineering is concerned with the engineering design of equipment and soldier-machine interface so that system performance, including the human element, is maximized. This issue is, to a large extent, synonymous with workload. If the interface has been designed well, the ease with which the maintainer and operator can perceive information or perform tasks may be optimal, and performance will be high. A poorly designed interface will be an important factor in increasing workload, accompanied by potential performance decrements and possible mission failure.

2.2 Overview of MANPRINT Method Products

Six products are being developed to support effective implementation of the Army's MANPRINT Methods Program. Throughout the remainder of this report, we will generally refer to these six products as MMP1 through MMP6. Figure 2-1 depicts the relations among the six products and their relation to system design. MMPs 1 through 4 will assist analysts in the **pre-design** phase of system development to consider mission requirements, manpower, personnel, and training factors which interact to affect system design. MMPs 5 and 6 will focus on **evaluation** of designs and will feed information back to the design process if a system does not meet proposed mission performance criteria, considering the jobs to be performed and the personnel available. In summary, the six products and their purposes are:

- MMP1 -- provides support to TRADOC Combat Developers in defining missions, **system performance requirements**, and minimally acceptable performance criteria.
- MMP2 -- provides support to DCSPER and TRADOC to determine maximum allowable **manpower** requirements including crew size and maintenance manpower constraints.
- MMP3 -- provides information on the characteristics of all available Army **personnel** including aptitude, anthropometry, specialty, and skill levels.
- MMP4 -- provides information on the **training** available for system personnel and any training constraints.
- MMP5 -- **predicts the jobs** that will be required to operate and maintain the system and the number of personnel needed for each job.
- MMP6 -- **predicts the required characteristics** of maintenance and operational personnel based on a system design and mission performance criteria.

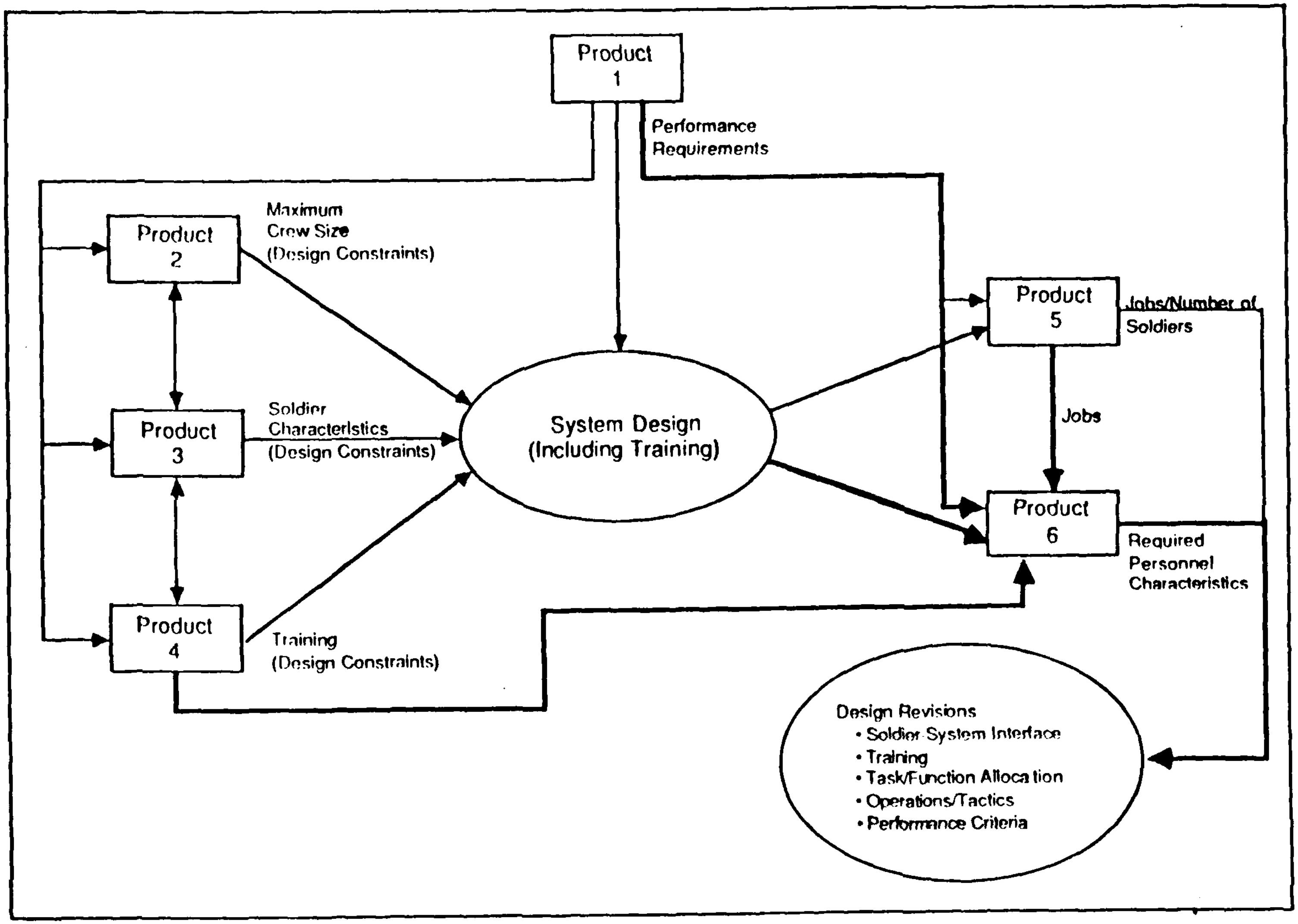

Product 1
Performance Requirements
Product 2
Maximum Crew Size (Design Constraints)
Product 3
Soldier Characteristics (Design Constraints)
Product 4
Training (Design Constraints)
System Design (Including Training)
Product 5
Jobs/Number of Soldiers
Jobs
Product 6
Required Personnel Characteristics
Design Revisions
• Soldier System Interface
• Training
• Task/Function Allocation
• Operations/Tactics
• Performance Criteria

Figure 2-1: Relationships among Products and System Design with Emphasis on Product 6.

MMP1 plays a key role in providing information about mission performance criteria to all other products. The four pre-design products (MMP1 through MMP4) interact with each other by sharing data and by determining limitations in the areas of manpower, personnel, and training that can affect a system design. Whereas limitations in any one area can be independently derived, an analysis of all factors simultaneously can sometimes facilitate the resolution of an apparent limitation in one area. For example, manpower limitations may suggest increased use of decision aids however, this may require higher caliber personnel. Similarly, if personnel characteristics are determined to be such that the system can be designed for personnel with a relatively high level of a certain aptitude, then potential constraints caused by training limitations might be eliminated.

2.3 Identification of Intended User of MMP6

The Life Cycle System Management Model (LCSMM) of the Army's process for system acquisition comprises five phases:

1) Pre-Concept Exploration
2) Concept Exploration
3) Demonstration and Validation
4) Full Scale Development
5) Production and Deployment

Use of MMP6 will be focused primarily in Phase 2, Concept Exploration (or, alternatively, on the Proof of Principle phase of the Army Streamlined Acquisition Process, ASAP). In this phase, an actual system design concept will be available to which MMP6 can be applied. In Phase 1, use of MMP6 will be limited primarily to preliminary work on the development of the simulation models required for later MMP6 application. In Phases 3. 4, and 5, MMP6 will be used primarily to validate previous applications, to evaluate any changes in design, and to provide guidance for technical and operational testing. MMP6 may also be used for extrapolation beyond conditions of testing and as a prelude to another round of Phase 1 concept exploration efforts.

The responsibility for using MMP6 in Phases 1 and 2 will rest primarily with the Combat Developer (CD) represented by the TRADOC System Manager (TSM). The TSM/CD will be advised by the MANPRINT Joint Working Group (MJWG) composed of representatives from several agencies which will typically include:

- TRADOC Proponent School/Center (Combat Developer),
- Army Materiel Command, and
- Soldier Support Center.

The MJWG will be responsible for the System MANPRINT Management Plan (SMMP), a "living" document that guides the application of MANPRINT throughout the acquisition process. While MJWG will not actually apply MMP6, MJWG members must be thoroughly versed in what MMP6 can do and what inputs are required. This group will determine the extent to which MMP6 will be applied and will schedule the application of MMP6 to coordinate with other MANPRINT efforts (e.g., applications of MMPs 1 through 5) and the total acquisition process. It will also identify resource requirements for application of MMP6. The MJWG will adjust and guide the application of MANPRINT Methods (including MMP6) throughout all phases of the LCSMM.

After Milestone 1, (completion of Phases 1 and 2) overall responsibility for acquisition, and therefore for MANPRINT, shifts to the Materiel Developer (MD), usually at AMC. The MD and the TSM may also each have a staff officer with designated responsibility for MANPRINT. These officers, as well as the members of the MJWG, will typically be people with some experience in manpower, personnel, and ~~training~~ (MPT) but without expertise in simulation of human performance. Because of the special expertise required to develop human-system simulation models, the actual application of MMP6 ~~should generally be~~ performed by a group with appropriate skills in task analysis, systems analysis, and simulation. Such a group could be either internal to the Army (e.g., the Army Research Institute for the Behavioral and Social Sciences and the Army Human Engineering Laboratory have people with appropriate expertise) or the user could be an outside contractor.

In addition to the primary users identified above, other users of MMP6 might include:

- Deputy Chief of Staff, Personnel (DCSPER) -- DCSPER regularly reviews MANPRINT related progress in system acquisition including the output of MMP6. This review occurs throughout all phases of the LCSMM.
- Army organizations responsible for system test and evaluation -- These include the Operational Test and Evaluation Agency (OTEA), the Test and Evaluation Command (TECOM), and the Army Development and Employment Agency (ADEA). These organizations can use the output of MMP6 to point to specific tests that should be conducted and data that should be collected using other techniques. Also, the testing carried out by these agencies can be designed to support validation of MMP6 results and so reduce MMP6 testing requirements.

Detailed communications requirements for MMP6 will be determined in part by who uses the other MMPs. For example, the primary direct MMP6 user (assumed to be, for example, ARI, HEL, or a contractor) must receive information, particularly with respect to missions, conditions, and performance standards with regard to functions and tasks, from the user of MMP1. This MMP1 user will presumably be a staff officer for the CD, although possibly an officer different from the one responsible for MANPRINT. Early transmission of detailed MMP1 information to the MMP6 user will facilitate the process of simulation development.

Additional key information coming into MMP6 will come from MMP5. Efficient communication will be critically important here, because both products will be used in a similar time frame, primarily during Phase 2 of the LCSMM. The critical information from MMP5 will be task analysis data. These data are needed to define the simulation models for system maintainers and operators. Useful task analysis data may also be available from other sources (e.g., the ISD process which supports training development). Thus, MMP1 will provide "the what" - mission performance by function and task; MMP5 will provide "the who" - job descriptions.

The outputs and results of applying MMP6 will go primarily to three types of users:

- System designers
- System testers
- Reviewers/decision makers

System designers include AMC Program Managers, TRADOC System Managers, and contractors; system testers include OTEA, TECOM, and ADEA; and reviewers/decision makers include DCSPER and review bodies such as the In-Progress Review (IPR), the Army Systems Acquisition Review Council (ASARC), and the Joint Management Review Board (JMRB).

2.4 Objectives of MMP6

The main body of this report addresses the development of a design for MMP6, which is a software tool for System Evaluation for Personnel and Training Analysis (SEPTA). SEPTA is an evaluation tool which predicts what personnel characteristics are required given a particular system design, specific mission requirements, and job design. SEPTA will interact with other products, both by using data derived from the other products and also by returning data to the design tools. Figure 2-2 focuses on MMP6 inputs and outputs. As illustrated in Figure 2-2, the major inputs to MMP6 are the Product 1 system performance requirements, the Product 5 job descriptions, and the system design itself. The principal outputs of SEPTA are detailed descriptions of personnel characteristics that are required for successful operation of the system of interest. The more detailed performance prediction outputs of SEPTA will also make it particularly useful for system development issues in the areas of function allocation, assessments of the impact of training on performance, development of tactics for use of the system, and comparison of alternative user-interface designs.

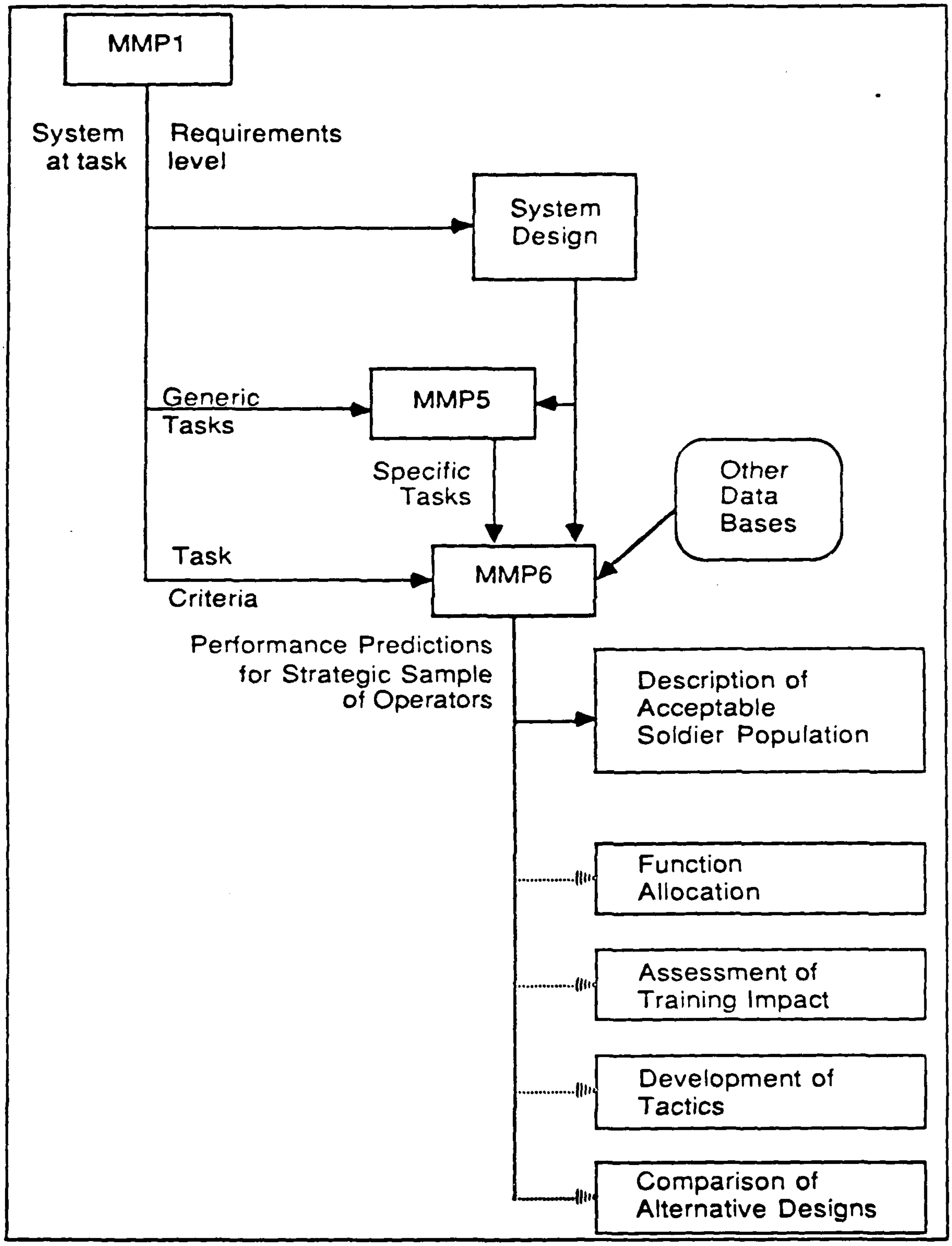

MMP1
System at task
Requirements level
System Design
Generic Tasks
MMP5
Specific Tasks
Other Data Bases
Task Criteria
MMP6
Performance Predictions for Strategic Sample of Operators
Description of Acceptable Soldier Population
Function Allocation
Assessment of Training Impact
Development of Tactics
Comparison of Alternative Designs

Figure 2-2: Schematic Diagram of Product Relations for MMP1, MMP5, and MMP6

The personnel characteristics output by SEPTA must be represented in terms that are suitable for the intended user of that information. The user might wish to consider these outputs in conjunction with the soldier population characteristics projected by MMP3, for example. The user might also want to relate the SEPTA outputs to various other tools and data bases. Accordingly, it will be desirable for SEPTA to offer the user a fair degree of flexibility in the specification of outputs, with options ranging from detailed profiles of personnel characteristics over many variables to succinct measures of the portion of a target soldier population that is accommodated by a system design.

SEPTA will accomplish its objective by considering the separate capabilities that will be required to accomplish each job. However, the method used to evaluate the required personnel capabilities will account for the multivariate tendency of various capabilities to be clustered or correlated with each other. SEPTA is based on the recognition that most capabilities are not distributed independently in the population.

The overall scope of SEPTA is intended to be quite broad. It is intended:

- To apply to all types of military systems,
- To address the performance of both maintainers and operators,
- To provide a means for considering training issues, and
- To incorporate and identify the positive and negative effects of all relevant performance factors.

Several major innovations in performance prediction technology are required in order to achieve this objective; these will be described in terms of SEPTA requirements, current technology status, and a plan for developing and validating the required capabilities.

The objective of dealing with both operational and maintenance personnel actually appears to be a somewhat moot issue considering the extensive overlap of behaviors involved in these two categories. Maintainers are increasingly performing their tasks via workstations and procedures that are very similar to those used by system operators. Indeed, the maintainers may sometimes be considered as operators of the maintenance systems. At the same time, there are still many tasks performed by system operators that are manually intensive (e.g., assembling and disassembling field equipment). Accordingly, we will not attempt to distinguish strongly between operators and maintainers throughout the remainder of this report; we consider the full range of tasks performed by either maintainers or operators of Army systems as appropriate objects of SEPTA application. In general, wherever we refer to "human performance" or even "operator performance," we intend to include the performance of both operators and maintainers.

It should be clear that the objectives of MMP6 and SEPTA are extremely ambitious. We additionally provide for possible future additions and capabilities which are beyond the scope of the statement of work. Successful development of SEPTA will constitute a major advance in systems evaluation technology. Although simulations are now occasionally used to evaluate system designs, these efforts rarely devote consideration to the differentiation between capabilities vs performance of the human maintainers and operators. In the few cases where just human performance is considered, it is only nominal or average performance that is addressed and not the variability in performance that is associated with differences between humans in individual characteristics and abilities. Thus, SEPTA will be required to break new ground in the simulation of how individual differences in human abilities and characteristics will impact on human-system performance. We will, in the next few paragraphs, examine some problem areas for SEPTA design and development that are implied by the MMP6 objectives.

One apparent difficulty encountered in accomplishing the objective of SEPTA results from the fact that the desired prediction is in the reverse direction from the common notion of causality and, therefore, from the direction provided by most available models. Traditionally, system design and human capabilities are viewed as inputs and actual performance as the output; this will work if the required human capabilities are known beforehand. In the case of SEPTA, however, the purpose is to determine the human capabilities required by the system where the required performance is defined as part of the system requirements. **Thus, the traditional logic is reversed: performance is an input and human capability characteristics are an output.**

Another difficulty is that of isolating the various influences on system performance to determine their impact in evaluating personnel characteristics. Figure 2-3 illustrates this relation in Venn diagram form. It simultaneously illustrates some of the mapping required for MMP6, and consequently SEPTA. The outer smaller circles represent contributing factors and correspond to scenarios, environment, personnel characteristics of maintainers and operators, and other platforms. Overlap between these small circles indicates some of interactions which may take place among these components; for example, a cold environment will slow maintenance. Overall performance is depicted in the large circle labeled System Performance. Space in the large center circle not covered or overlapped by the outer circles can be considered to represent the sole contribution of system design, e.g., the weapons system being sufficient to accomplish the goal and requirements. One purpose of SEPTA is to isolate the influence of the various influences to determine the degree of overlap (shaded areas of influence) of the contributing factors.

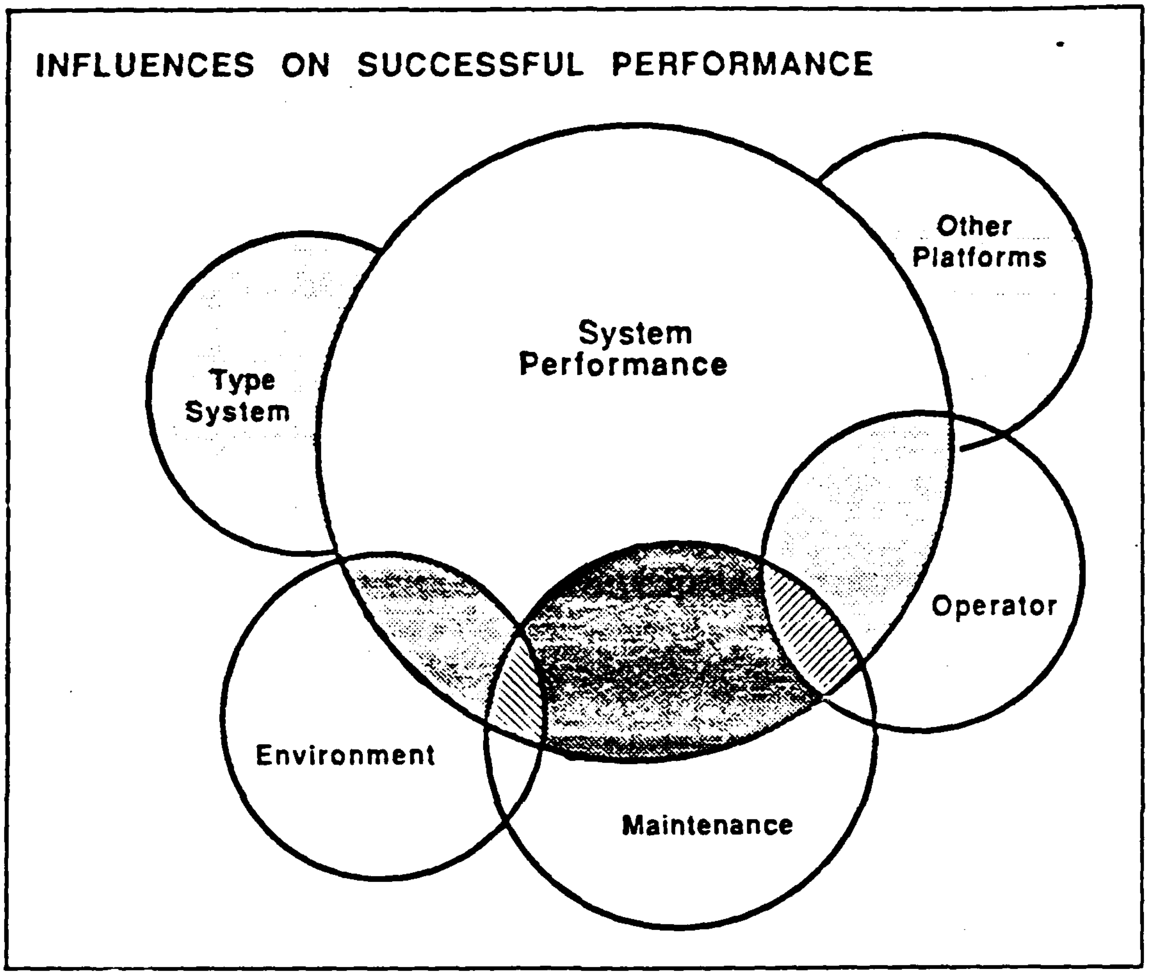


Figure 2-3: Illustration of the influences on successful system performance.

Another area of difficulty arises from the complexity of the constraints associated with some of the modeled variables. SEPTA is being designed to be used in conjunction with other MANPRINT tools and with a variety of established data bases. In particular, SEPTA will obtain a definition of system requirements from MANPRINT Methods Product 1 (MMP1) which is used to derive detailed system requirements from higher level, more general considerations. These system requirements will be defined in terms of speed and accuracy requirements for specified tasks. Since MMP5 will be used to identify the specific job positions which will be required to support, maintain, and operate proposed Army systems and to establish the functions and responsibilities associated with each job, the outputs of that product will provide the basis for the simulation of job performance in SEPTA. Thus, MMP1 and MMP5 will provide the important data for the accomplishment of the SEPTA objective, and in doing so, they also impose significant constraints on both the user-interface and simulation techniques used in SEPTA.

(There are many ways of interacting among products. For example, SEPTA might also interface conceptually with the results of MMP3 which describes personnel characteristics of the soldier population which is expected to be available for manning Army systems in the future. This connection would be particularly strong if MMP3 were to incorporate a detailed database of personnel characteristics that could be projected to various future times of interest (but this capability is not a required characteristic of MMP3). More likely, SEPTA will interact with the training outputs of MMP4. In particular, we will be able to identify and distinguish among performance components which can be trained and those which cannot.)

Probably the most serious difficulty encountered in the design of SEPTA is the need to address a broad range of types of cognitive activity in addition to manual, perceptual, and psychomotor performance. This is the issue of feasibility. Special problems arise in predicting cognitive performance; it is much less observable and the mechanisms by which it operates are less well understood than are non-cognitive aspects of performance. Cognitive tasks, in addition, generally admit a much greater variety of alternative behaviors that would be considered appropriate or successful than in the case of non-cognitive behaviors. For example, there are relatively few plausible ways for an individual to lift an object, detect a visual target, or track a moving symbol. There is frequently much greater latitude in viable approaches to solving a problem, making a decision, or other cognitive activities. For example, research has shown several modes of learning a path through a maze, involving either spatial or verbal (LRRLL...) memory schemes.

Feasibility is too broad an issue to be dealt with here. However, we have developed workable techniques to deal with this issue. Some discussion in the concept report (Glenn, Dick, & Bittner, !987) addresses this issue. Considerable, additional background work and effort has been directed at this problem. An analysis of available Project A data is appended which reflects and substantiates the approach. Additional material will be made available detailing and substantiating the approach.

2.5 Limitations of Scope

Remaining efforts under Task 3 of this project will construct an initial software version of SEPTA which will incorporate the important features required for the tool and so provide a useful instrument for system design evaluation. At the same time, however, this initial version of SEPTA will not include all features that might ultimately be beneficial and possible to implement. There are, of course, many software features which could be added to make SEPTA even more useful. In particular, the initial version will consider individual differences in body dimensions and cognitive and psychomotor capabilities, but will only begin to deal with the factors of sensory and strength differences between people.

At a different level, some influences on performance will be ignored initially. For example, the initial version will ignore the effects on human performance of stress and motivation. Because of our modular design approach, however, all of these factors can be introduced into SEPTA at a later time with no additional cost because of the delayed introduction. Similarly, detailed connections with MMP4 (training) are also omitted as part of the design, although SEPTA results will have implications for training. This planned staging for the development of SEPTA is based on the observation that cognitive, psychomotor, and anthropometric variables usually account for the majority of personnel characteristic variance.

Having considered the broad scope, the role of SEPTA in the organization, we turn to lay the ground work for the specific design presented.

3. SEPTA CONCEPT

The approach taken in the development of this tool is quite different both by necessity and purpose from more "traditional" job analysis approaches and much more bold. The purpose of traditional approaches is to identify the tasks and personnel characteristics associated with a particular job. Our purpose is to identify the **required level** of personnel characteristics associated with a job or package of duties and tasks. Clearly, our purpose assumes the former, namely, the ability to identify the tasks and personnel characteristics. Since we build on the typical job analysis theory approach it will be useful to consider briefly what approaches have been taken and what is known.

3.1 Traditional Job Analysis Approaches

Job analysis is typically performed by asking people who are familiar with the job to make subjective associations between job elements and the component personnel characteristics that are required to accomplish each element. Jobs may be analyzed to various different degrees before they are mapped into required personnel characteristics. For some purposes, the analysis may be performed at the total job level with no discrimination at lower task levels and responsibilities; this approach is generally reserved for managerial and professional jobs which involve diffuse responsibilities and broadly based tasks which cannot be cleanly defined. Most commonly, the job is broken into component tasks and the job expert is asked to estimate the level of each personnel characteristic that is minimally necessary to perform the task. In some cases, a fairly thorough task analysis may be performed to identify elementary activities that comprise each task. From the analysis, personnel requirements are estimated by experts for each activity.

A variety of different taxonomies of personnel characteristics have been used to support job analysis methods. Fleishman (1967, 1972) has used factor analysis to identify the dimensions of skilled performance (initially with a focus on psychomotor and physical proficiency and more recently with extensions to address cognitive aspects of performance). Guilford has used a similar approach to develop a model for the structure of cognitive functions (Guilford, 1967; Guilford & Hoepfner, 1971), though Guilford started with a conceptual model which he sought to validate through factor analysis of performance data. Similar approaches have been used in the analysis of the acquisition of skill (Adams, 1987). (More recently, a new theoretical structure and a new hierarchical factor analysis [Wherry Sr., 1984] has been used to identify basic underlying internal processes of cognitive capability associated with reading and interpreting typical tactical displays [Wherry Jr., 1985; 1986; 1987]. Although promising, this recent effort has yet to be applied to a wide range of tasks.) A widely used job analysis instrument known as the Position Analysis Questionnaire (PAQ) (McCormick, 1974) employs questions about a large number (187) of job elements covering information input, mental processes, work output, relationships with other persons, job context, and other job characteristics; results obtained with the PAQ have been factor analyzed to reveal independent dimensions (Marquardt & McCormick, 1974).

There are two interrelated problems with these traditional approaches to job analysis – subjectivity and uncertainty. Subjectivity is a problem because people are frequently and unknowingly biased and unreliable in subjective estimation tasks, hence introducing uncertainty into the results. (This is true especially when a series of subjective estimates have to be made, that is, when estimates are based on other estimates.) The subjective approach poses a further problem when it is desired to analyze jobs and systems which do not yet exist (a typical condition for MANPRINT applications) and for which there are really no job experts. Even if the subjective estimates of task requirements were not suspect, there are other troubling uncertainties in these traditional job analysis methods. These analyses typically indicate an average level of each personnel characteristic required but not how **critically** it is required. Consequently, considerable uncertainty remains concerning the degree to which performance would be degraded if personnel with sub-criterion characteristic values were to attempt the jobs. It is important to recognize that this uncertainty would be further exacerbated in the MANPRINT application in the process of translating basic characteristics descriptions into MOS descriptions. Evaluation of the relationship between individual abilities and system performance requires a stronger, human performance model-based methodology.

A somewhat different approach to job analysis involves developing human performance requirements specifically for a system. These originally took the form of time-line analysis but more recent efforts have evolved into task analysis which is used in the attempt to define and identify the overall workload. Table 3-1 illustrates the structure of task analysis according to Army recommendations. This structured task analysis technique represents an important methodological step beyond time-line analysis through the required specificity and relation to mission goals. Some of the recently developed techniques classify and identify performance in four categories, visual, auditory, cognitive, and psychomotor (McCraken & Aldrich, 1984). Usually, times for the task and numerical ratings for difficulty are assigned to each of the four

Table 3-1: Description of Components of a Task Analysis (From: Analysis Handbook (TRADOC-Pam-351-4(T) and Task Inventory and Task Analysis (MIL-HDBK-XXXX, proposed)

General System Components

Mission: Statement of the goals and what the mission is supposed to accomplish.

Scenario/Conditions: Factors or constraints under which the system is expected to operate, e.g., nighttime or a restriction on units available for action.

Function: A general statement of the activity performed by the system, e.g., ground transportation.

Specific Personnel Components

Jobs: General statement of the performance required of each person in the system.

Duties: A set of operationally related tasks within a job.

Task: A set of related activities of human performance such as perceptions, decisions, responses for a specific purpose, e.g., communicate. e.g. Change a tire. (Broken into critical and non-critical tasks.) Often composed into a task inventory which is a comprehensive listing of all tasks performed by system personnel.

Critical Tasks: Those tasks which if not done in accordance with system requirements, will have an adverse effect on criterion effectiveness, reliability, efficiency and cost.

Other tasks: Those which are non-critical tasks.

Subtasks: Activities breakdown of the task into smaller units, e.g., remove lug nuts.

Task Elements: The smallest unit of behavior required, e.g., apply counterclockwise torque to the lug nut with a lug wrench. (Rarely entered into a task inventory.)

categories separately and the assumption is made that the four categories are independent. Many difficulties remain, however. For example the form of information presented in a display can have important consequences on performance. Analog displayed information is usually easier to perceive than digital information (Harris & Glover, 1984) and this difference can have important consequences for mission success.

One aspect of the of the importance of a task analysis as illustrated in Table 3-1 is the detail required. The requirements force the user to describe fairly detailed requirements of the job. Another important aspect is the mode of thinking in terms of dissecting performance into smaller components. (This latter component is not trivial in the context of institutionalizing SEPTA. It will be very useful to extend the mode of thinking rather than requiring a different mode.) SEPTA combines both of these points. Loosely conceived, the dynamic simulation in SEPTA is an extension of the task analysis technique. An important extension is that the task elements are rigorously and objectively defined. However, in order to predict performance adequately in key situations, one has to consider detailed performance as a consequence of interacting components. Although software design is the foremost issue in developing SEPTA, design of the human model vis a vis micromodels and their interactions is also important. This is the critical point of departure from traditional and especially task analysis methods: task analysis is time-line based and network oriented, SEPTA is event and interaction oriented. Sticha (1987) has stated the latter approach is more powerful and flexible.

3.2 Our Approach – Theory of Operation

SEPTA builds on information available from other products and sources. To start with, there are data available on individuals, e.g., from Project A, and data on tasks, e.g., from Product 1 (MMP1), and job clustering from Product 5 (MMP5). SEPTA builds on these data and initially uses them "as they come" - the user is not required to make (or have made for him) any additional subjective estimates. On the results end, we obtain; a) detailed specifications for individuals required to perform the tasks and jobs; b) detailed specifications of the tasks themselves described in a form which was not available at the start; and c) suggestions for training as a result of knowing the characteristics of the individual and task specifications.

Figure 3-1 shows the flow of information into SEPTA and outcomes as a consequence of the analysis capabilities of SEPTA. The abilities required for systems maintenance and operation represented in the upper left are actually contained in specifications of tasks and desired system

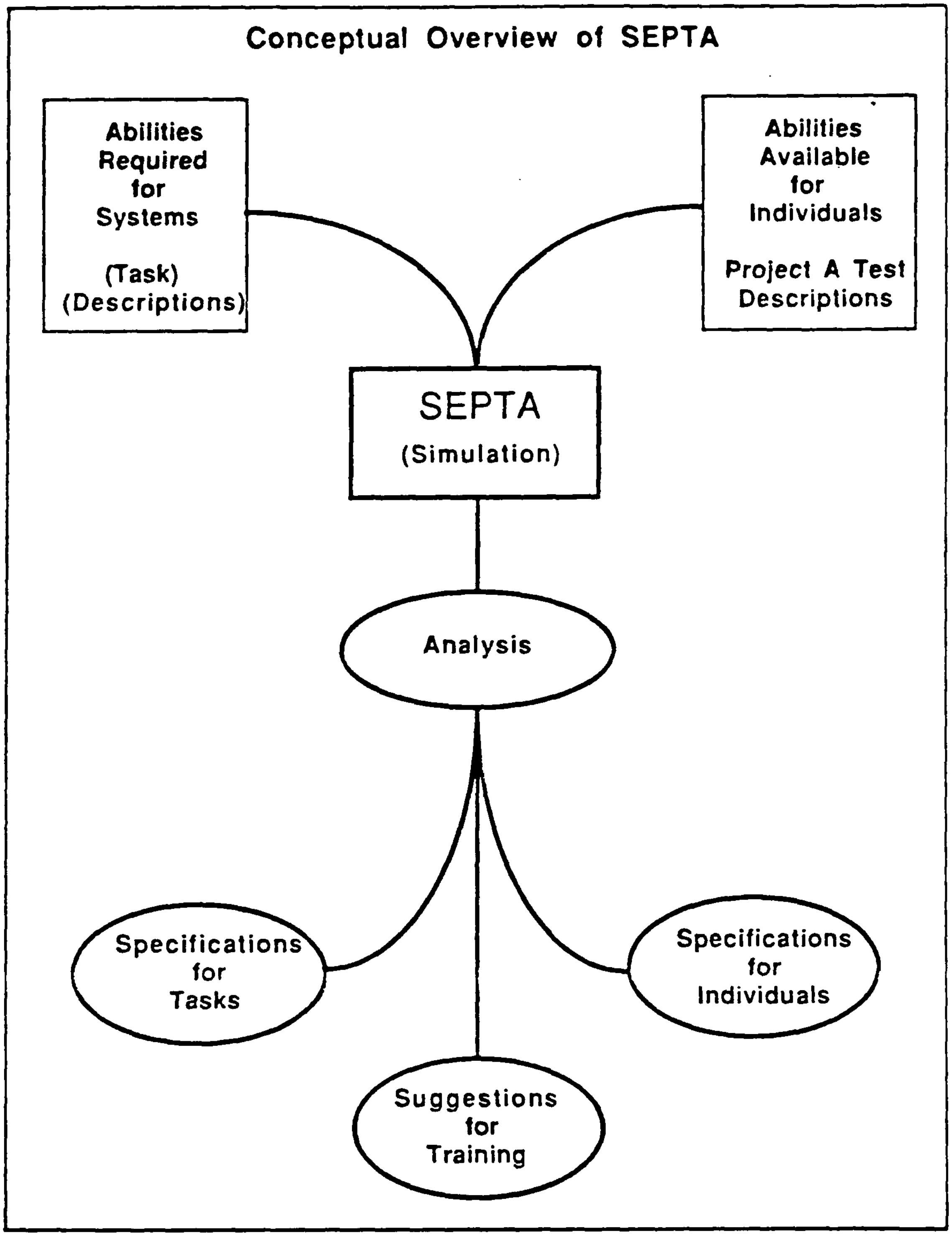

Conceptual Overview of SEPTA
Abilities Required for Systems
(Task) (Descriptions)
Abilities Available for Individuals
Project A Test Descriptions
SEPTA
(Simulation)
Analysis
Specifications for Tasks
Specifications for Individuals
Suggestions for Training

Figure 3-1: Schematic Representation of the Flow of Information into SEPTA with Resulting Outcomes.

performance in the form of time and accuracy from MMP1. (E.g., the operator should identify a target and destroy it with x seconds with 98% accuracy. The maintainer should be able to change the spark plugs within x minutes.) The associated and required underlying abilities' will be "unearthed" in the application of SEPTA. On the upper right we have represented the measured and known abilities of soldiers. These are most typically and usefully represented in personnel tests of various kinds, such as those in Project A and other data. Project A represents a relatively broad range of perceptual, cognitive, and motor abilities, and data themselves are representative of a broad distribution of individuals.

We convert these two data sets (task descriptions and test descriptions) into a common form and bring them together in SEPTA. Through simulation we obtain results. Through analysis and, by reversing the conversion process, we provide the specifications and suggestions illustrated at the bottom of the figure. That is, we have very detailed, objective task specifications. These will often be of general use to designers by illustrating the subtle consequences of presumed innocent interface changes. We will also have as detailed and objective as possible specifications for individuals rated in terms of the relative importance of the necessary levels of abilities required to perform the tasks. Finally, we will be able to differentiate between those capabilities which can be enhanced by training such as strategies and those which are difficult or impossible to enhance such as decision time and reaction time. This latter point will have training implications.

Types of Models

Analytic procedures for the evaluation of a new system can take a variety of forms. (See Lysaght et al., [1987] for a complete review of analytic techniques.) The dynamic simulation offers the flexibility the modeler needs. Recently, Sticha (1987) has discussed two general techniques to simulate human performance. According to Sticha, the difference between these two existing classes can be stated in terms of the ways in which the control of sequencing of the behaviors is accomplished. The first of these is a network model as represented by SAINT and Micro SAINT. This approach controls the order directly in a network by means of the way the analyst has developed the procedures - **order is defined in the procedure**. Network models are a combination and amalgamation of a number of techniques: flowcharts, program evaluation and review technique (PERT), Markov models, decision trees, and reliability models. The second method of simulation controls the ordering through a set of rules for production and through these rules by the environment. Sequencing is indirectly inferred by a set of rules that associates a behavioral action with an environmental event. The actions are performed only when the environmental conditions of the rules have been met - **order is defined by the environment.**

While these models may in some situations produce identical results they can have different capabilities and it is reasonable to clarify the design approach taken here. Sticha points out that procedural tasks are characterized by internal control whereas tasks involving the recall and application of rules are driven by the environment. Sticha has classified our dynamic simulation as a network model, an accurate classification for previous versions but not current version or for the extensions envisaged for the present purposes. HOS-IV permits both types of rules; a time-line is the "degenerate" case for HOS. This means the output and evaluation of the results can vary from a simple time-line through detailed performance characteristics. Indeed, as we discuss the design of the micromodels, it will become clear that both the procedural and the production ideas are incorporated. Note: The use of the term 'procedure' in the program description sections is in the context of computer languages and subroutines, not modeling.

In particular, the "engine" of SEPTA is the dynamic Human Operator Simulator (HOS-IV) which is a human simulation concept developed over a period of years. (See the bibliography for references.) HOS-IV is a useful tool for a broad range of system development applications. It can be applied to any man-machine system in which operator tasks and system functions can be applied analytically. HOS-IV has been developed to be used early in systems design to evaluate function allocation and systems configuration. When defining a simulation, the user can start at a high level of detail and, for example, simulate only discrete events to gain an understanding of the environment and scenario. The user can then add systems models to determine the hardware system processing rate. Subsequently the user can define the tactics and procedures which the operator would use in executing the mission. Following simulation definition and execution, alternative designs can be determined and the simulation rerun to evaluate the impact of alternative designs on performance. The exercise of defining the procedures required to be performed for a task can yield valuable insights for the designer without actually running the model. It encourages the designer and analyst to think through the tasks and can be used to determine consistency and completeness of micromodels. Developing a HOS simulation thus provides a discipline and consistency to the design process.

The Rest of the Report

In the next sections, we describe the necessary software and the associated design required to accomplish the outcomes suggested in Figure 3-1. In Section 4, the general user interface with SEPTA is described. In Section 5, the program design and specifications for setting up simulations are described, as well as the design of necessary data bases, and statistical concepts. The dynamic simulation is described in Section 6 and the static simulation in Section 7. Analysis of the simulation results and conversion into personnel characteristics is described in Section 8. In Section 9, subroutines and other tools are considered.

4. GENERAL DESCRIPTION AND USER INTERFACE

The system is composed of five main parts: Setup, Dynamic and Static Simulation, Analysis of dynamic results, and Reports of the simulations. These parts will each be discussed in turn. Before that, however, we will consider the user interface and the ways the user will interact with SEPTA. In this section, the user interface will be outlined. References will be made to other, general system programs such as report generator, input formatter, editor, etc. as appropriate. Each of these will be discussed later. Figure 3-1 illustrates a conceptual flow for SEPTA. In this section we begin the process of explicating the detail design of SEPTA.

4.1 System Specifications

System specifications and general guidelines were developed according to the best thinking of the entire group of MMP developers taking into account Department of the Army requirements. These specifications are provided in Appendix A of this report. This effort also took into account integration of the various products in the MANPRINT Methods Series as well as limitations imposed by the hardware available and ARI specifications.

4.2 The User's Viewpoint

Screen design is an especially important basis for user acceptance. Accordingly, a considerable amount of our effort and thinking has already been expended in the attempt to make the system as easy for the user as possible. The basis and background of the screen design is presented in Appendix B. Throughout the design, there are a number of points and sometimes modules to help the user, to make it easier for the user, and to offer useful guidance. Because this is, in part, an evolutionary process, this effort will continue for as long as we are involved with SEPTA. The points will ultimately be put in the Users manual to offer helpful guidance in using the system.

4.3 System Profile Concept

The structure of a profile is the heart of the SEPTA design and of importance in the screen design. It represents a file of files which are needed for system evaluation by mission. The profile helps the user keep track of what he has already done and indicates to him what may need to be done. The profile will serve as a reference table both for system defined files and the user defined input and output for the various program modules as well as for his selection of the user desired output criteria. An example of what a profile would look like is shown in Table 4-1.

Table 4-1: Illustration of a Profile.

System Profile: Helo1

Module	File	Source
Criteria &	helo1.spc	MMP1
Jobs	helo1.job	MMP5
Hypothetical	helo1.dyn	HIGMAS.dyn
Individuals	helo1.sta	User defined
KUIP Values		KUIIPMAS.dyn
.		
.		
.		

As shown in the table, the profile contains information about each module, files needed for and from that module, as well as the source of the file. This is an example with only a few program modules defined. This example shows that Criteria and Jobs have been converted and are available with their respective sources shown as MMP1 and MMP5. Additionally, the hypothetical individuals have been defined for both the dynamic and static simulations. The dynamic source is the master file; the static source has been entered by the user. The KUIP parameters have not yet been pulled from the KUIP Master (KUIPMAS) file and this needs to be done.

A change in an input file for a particular system analysis would require only the overwriting of the old file name within the profile. This allows the user to change portions of the data quickly and easily, and be able to re-run profiles easily. There are some "resident " files, for example, the correlation matrices for soldier populations, which serve as "master" files. These master files are available to the user at the touch of a button or he can enter his own specific file. The masters are not part of the system profile but rather get copied into the profile at the user's request. Thus, the integrity of the master files is maintained.

The user can work with profiles in several ways In one, the user can load an entire profile via the Top Level Screen and execute the entire file in a "batch environment." Another way, involves the user selecting a specific profile and then selecting the specific program module from the Top Level screen. He can then run that module on the desired profile, either to create a file or to modify the file in the profile. An example of this would be the user selecting HIG via the Top Level Screen, and then from the HIG screen, define an input file and run just the HIG module. This approach of a profile allows the user to run an entire system analysis, and has the flexibility to run a single step within SEPTA.

4.4 Screen and Module Relations - General Considerations.

4.4.1 Screens/Module Relations

The screen layout consists of various screens arranged *conceptually in three levels*. No matter which level, the screen will indicate the calling position of the current level, that is, where the user is. Figure 4-1 represents the arrangement of the three levels.

The two items at the bottom, Data Editor and Report Generator deserve some special comment, being connected to and reachable from every module. For example, the user can edit module data from any level or define or redefine the desired output criteria for that module. Special, hidden files maintain the report requirements and definitions so that the user does not have to enter his requirements each time he enters a module.

The Top Level Screen is the first thing the user sees when he fires up the system. This is illustrated in Figure 4-2. It shows the modules available and the recommended flow of work for evaluation of a system. The user can run the entire system from this Top Level or he can move easily to one of the program modules at the second level. This is accomplished via a menu choice inside each second level module screen. The user should be able to move quickly and easily from module to module to work with, change, modify and save the various components associated with the module.

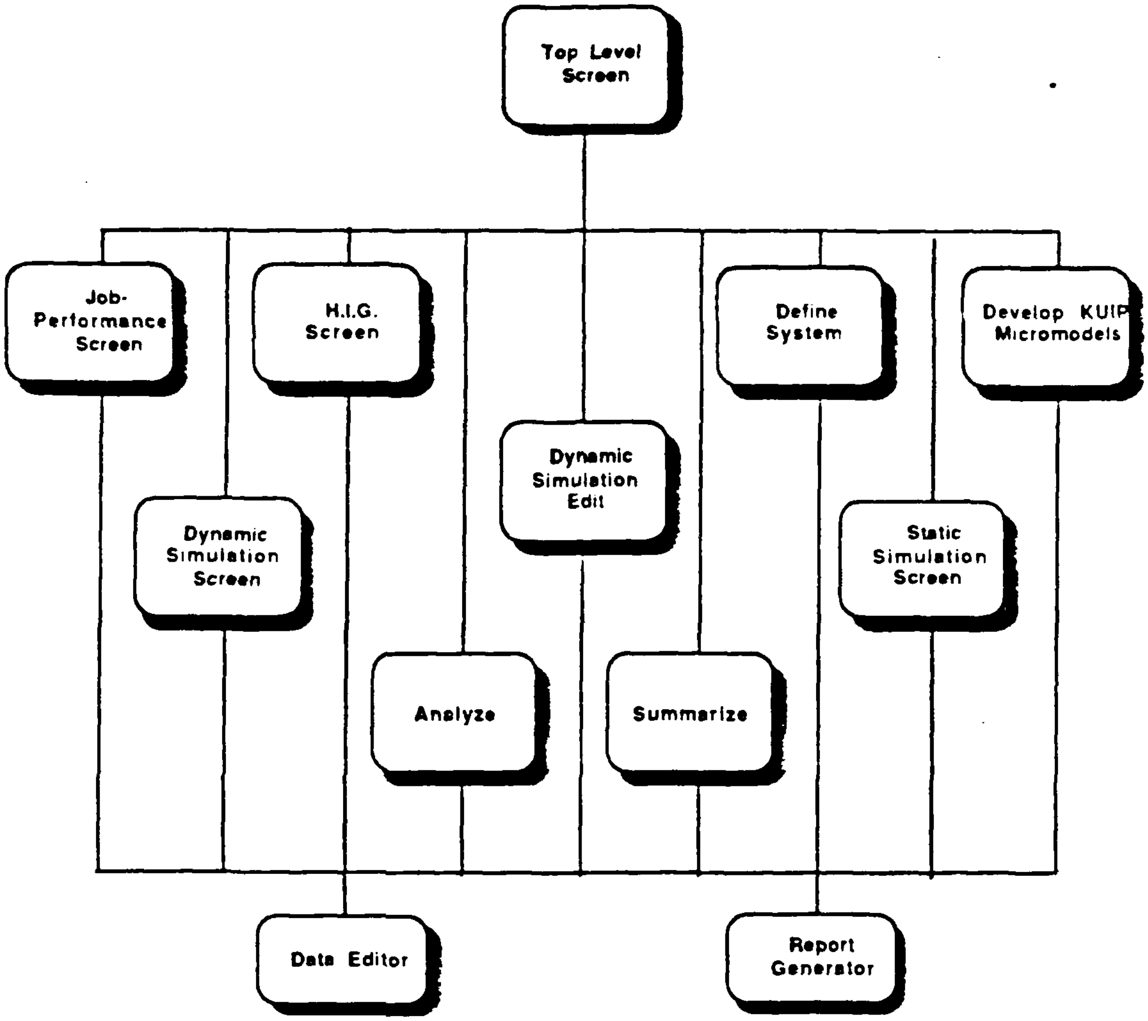
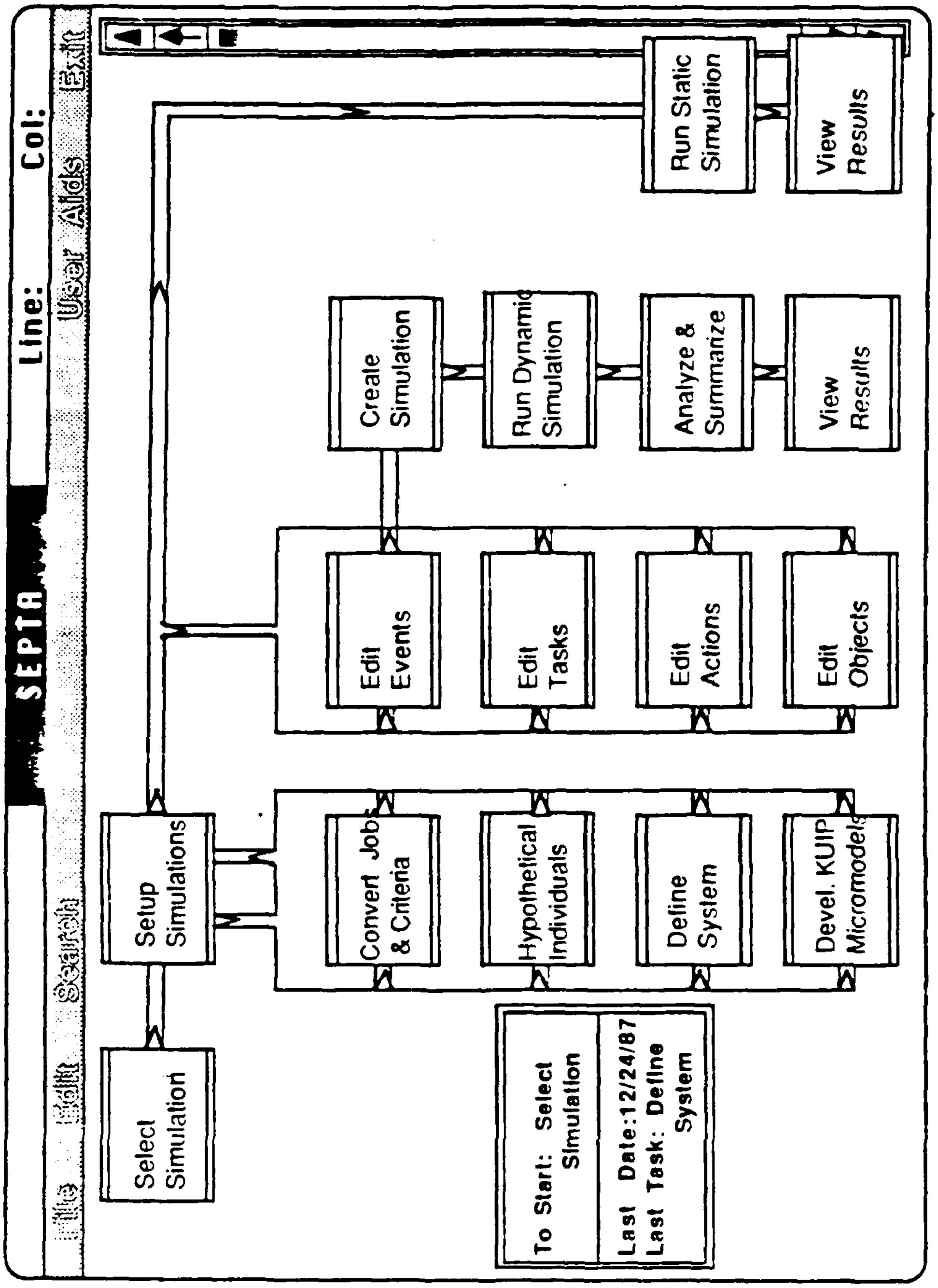


Figure 4-1: Illustration of the Arrangement of the Three Screen Levels.



To Start: Select Simulation

Last Date:12/24/87
Last Task: Define System

Figure 4-2: Illustration of the Top Level Screen Showing the Recommended Flow of Work.

4.4.2 General Choices

There are a number of options which will be uniformly available on each screen. These include:

4.4.2.1 Help Screens - User Aids

No matter where the user is, he will be able to get help to assist him in processing data and using the programs.

4.4.2.2 File Manipulations

Since it is possible for the user to add, delete, or otherwise change files outside the SEPTA system, it is appropriate to have options which permit the user to copy, rename, and use other MS-DOS functions without having to leave the system.

- Save
- Write
- Copy
- Directory
- Rename

Certain conventions are used throughout the SEPTA modules. All names are a maximum of 28 alphanumeric characters (a-z, 0-9). The first letter of each name must be an alphabetic (a-z). The only special character that can be included is an underscore (_). SEPTA is also case insensitive, that is upper and lower case characters can be used interchangeably. All names are converted by SEPTA into lower case for internal use. Examples of valid names are:

myname

my_sim_variable_name

abc123456

Examples of invalid names are:

a$	Contains an invalid special character ($)
a-b	Contains an invalid special character (-)
1qwerty	First letter is not alphabetic

4.4.2.3 Edit Text Files and Data Files

Both numerical and text files are used. The user will be able to edit any data file relevant at that point. (Actually, editing can be done from any module on any file.) Text files requests will spawn a text editor while R:BASE files will spawn an editor specifically designed for R:BASE files using calls to routines provided with R:BASE.

4.4.2.4 Run Programs

- Run current program
- Other choices
  Entire system
  Run Other programs - individually

4.5 The Detailed User Interface

The basic screen design for each level will be as similar as possible. The variations will be in the form of additional and specific menu options which will be used to set the various options for each specific program module. A number of screens will be shown illustrating various points about the user interface.

4.5.1 Screen Layout and Components

This section describes how information is arranged on the display and how the user interacts with the particular component. The screen is arranged into three main areas:

- The **title bar** which is always on the top line of the display as described in Section 4.5.1.1;
- The **menu bar** which is always the second line of the display as described in Section 4.5.1.2; and
- The **SEPTA window** which occupies the remainder of the screen. The SEPTA window is used to conduct a dialog with the user. It will either display information to the user or display an input form for the user to supply the information SEPTA requires. The contents vary dependant upon the current function. The windows are described in Sections 4.5.1.3 through 4.5.1.7.

Within the SEPTA window, a variety of components have been developed for the user to specify particular simulation data items or supplying additional information required before a system command can be processed. These components include:

- **List Selection Box**: a scrollable list of available items for the user to select from as described in Section 4.5.1.8;
- **List Viewing Box**: a scrollable list of currently defined terms for the user to view as described in Section 4.5.1.9;
- **Pushbutton**: a distinct area of the screen that is used to specify actions as described in Section 4.5.1.10;
- **Click Boxes**: a box containing the range of numeric values that can be modified by the user via mouse clicks as described in Section 4.5.1.11;
- **Text Entry Boxes**: a rectangular box which allows the user to enter textual data (numeric or alphanumeric) with the size of the box indicating the maximum number of characters permitted as described in Section 4.5.1.12,
- **Scroll Bar**: a rectangular box that is used to modify the current view of a window as described in Section 4.5.1.13; and
- **Labels**: text descriptions used to indicate the type of information to be entered by the user as described in Section 4.5.1.14.

SEPTA uses two distinct cursors to represent the focus of attention for the user and point to a precise point on the screen. The **mouse cursor** is controlled by the mouse and always represents the last mouse screen location. When the user moves the mouse, the mouse cursor moves proportionately. The mouse cursor is described in Section 4.5.1.15. Additionally, a **keyboard cursor** represents the location where any keyboard actions will occur and is described in Section 4.5.1.16.

In subsequent sections the following terminology is used to describe various aspects of the screen components:

- **Location:** indicates where the component is placed on the display. In some cases, the exact location will vary depending upon the contents of the screen;
- **Background Color:** indicates the color to be used for the screen background upon which text and or graphics will appear.
- **Text Color:** indicates the color to be used for all alphanumerics and text symbols;
- **Graphics Color:** indicates the color to be used for graphics symbols;
- **Border:** indicates the color to be used for the line border enclosing a particular element of the screen;
- **Characters:** indicate the case to be used for text, e.g., all upper case, initial upper case, etc.; and
- **User Action:** describes how the user will interact with the particular screen component.

4.5.1.1 Title Bar

The title bar presents information about the currently selected SEPTA function to the user; the user does not enter any information. The title bar is continuously displayed and is illustrated in Figure 4-3. The title bar is split into three areas:

- Current activity on left side of line, left justified with initial caps, if required. (Currently used by action editor.)
- Function name centered in middle of line in uppercase white letters on black background. Required for all SEPTA modules.
- Status information on right side of line with initial caps, if required. (Currently, used by action editor to display line count and column position.)

Location:	Top line of screen.
Background Color:	White
Text Color:	Black
Graphics Color:	None
Border:	None
Characters:	Varies depending upon area.
User Action:	Information only, no user response permitted in the title bar area of the screen.

4.5.1.2 Menu Bar

The menu bar contains a list of the menu titles of the primary options that are available for the current SEPTA function. The menu bar is continuously displayed on the screen and is illustrated in Figure 4-3. The last two menu options are always: User Aids and Exit. Each menu title is separated from other menu titles by one leading and two trailing spaces.

Location:	Second line on screen.
Background Color:	Blue
Text Color:	White
Graphics Color:	None
Border:	None
Characters:	Initial upper case only.
User Action:	To select an option from the menu bar, the mouse is used to position the mouse cursor anywhere on the desired menu title. Without moving the mouse, any mouse button is pressed and

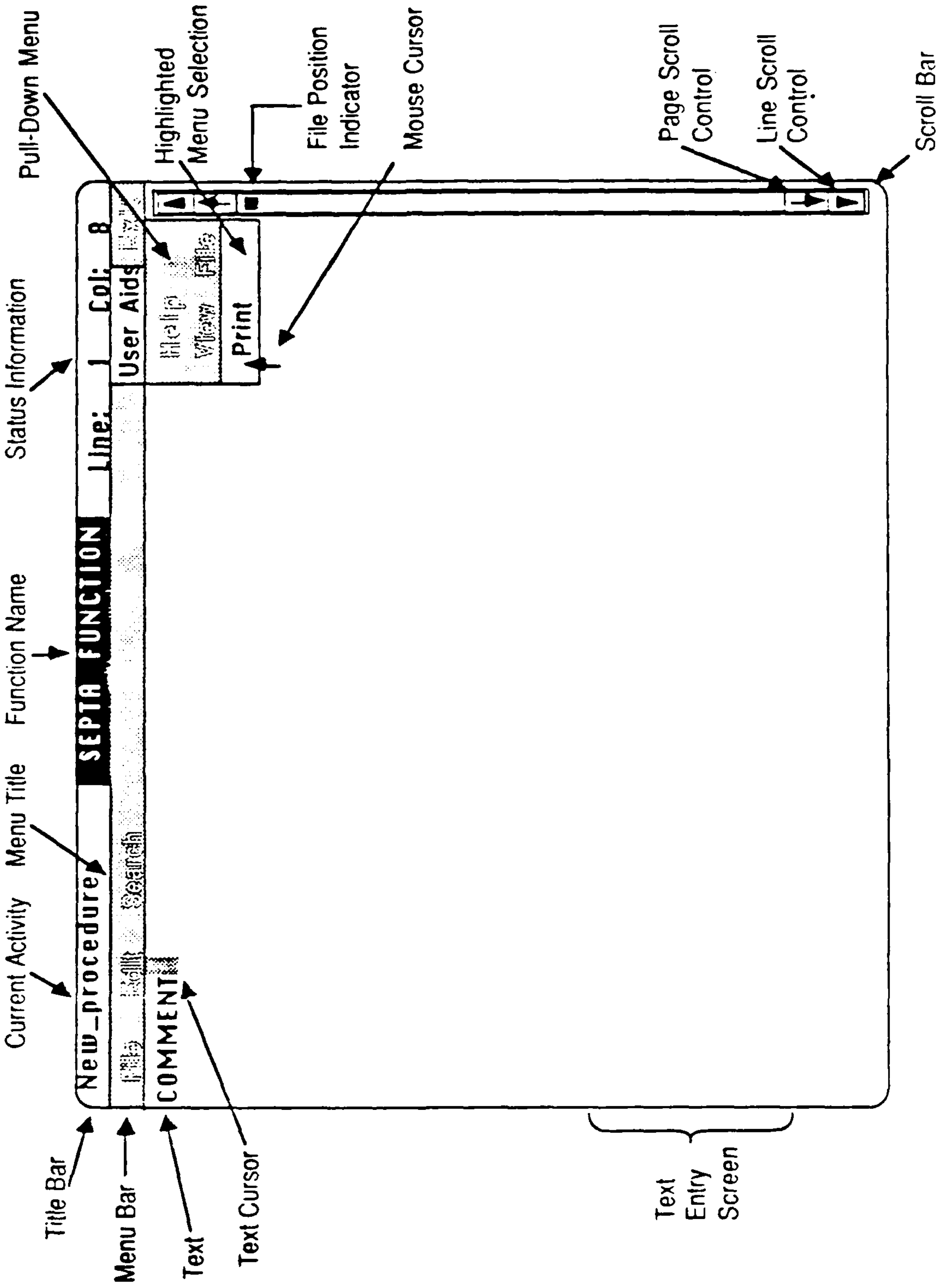


Figure 4-3: SEPTA Screen Annotated to Show the Various Components and Details.

held (clicking). Once the mouse button is depressed, the selected menu title will be shown in reverse video (white background and blue foreground) and a box containing the available commands (pull-down menu) will appear immediately beneath it in a separate window. The pull-down menu will disappear as soon as the mouse button is released. In order to view all the pull-down menus, the user can drag the mouse across the menu bar and as each menu title is selected, the accompanying pull-down menu will be displayed.

4.5.1.3 Pull-Down Menus

The pull-down menu is a separate rectangular window displayed beneath the menu bar containing the list of commands available for a particular menu title on the menu bar. It is illustrated in Figure 4-3. It is displayed only from the time the mouse button is held down and the mouse arrow is dragged down through the menu options until the mouse button is released. The pull-down menu window may obscure the previous contents of the screen while it is active.

Location: A separate rectangular window whose top is immediately beneath the menu bar and upper left corner is aligned with the selected menu title. The menu text is indented two spaces to the left of the selected menu title. The window is one space wider than the title of the longest command title.

Background Color: Blue

Text Color: White

Graphics Color: None

Border: None

Characters: Initial upper case only.

User Action: To choose one of the listed commands in the pull-down menu, the mouse is used to move the mouse pointer to the displayed menu title on the menu bar. While the mouse button is held down, the mouse is used to move the mouse pointer to the desired command (dragging). When the mouse pointer is located over the selected command, the mouse button is released. As the mouse pointer moves to each command line, the currently selected command is highlighted in reverse video (white foreground and blue background). The command that is highlighted when the mouse button is released is invoked and the pull-down menu disappears. If the mouse cursor is relocated within the menu bar line and the mouse button released, no action will occur and the pull-down menu will disappear. Similarly, if the mouse cursor is dragged outside of the pull-down menu window and released, the pull-down menu will disappear and no command will be chosen.

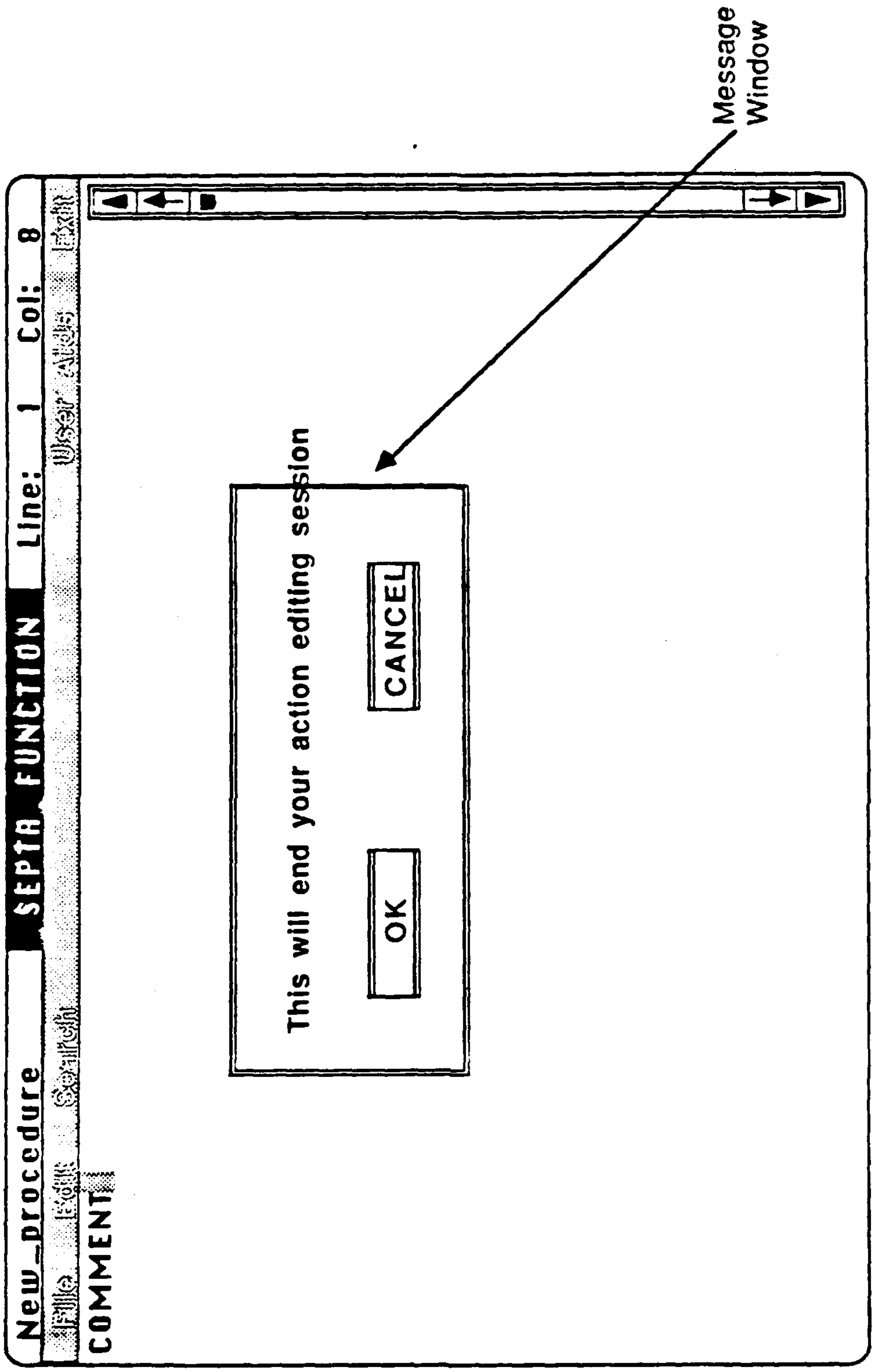
4.5.1.4 Message Windows

An informative message about the current system action requesting the user to indicate subsequent actions such as whether the currently selected action should occur or be cancelled. The options available to the user are displayed as pushbuttons and one of the pushbutton must contain a CANCEL option that permits the user to cancel the current request and resume the previous activity. The message window is illustrated in Figure 4-4.

Location:	A separate rectangular window that is located in the workspace beneath the menu bar.
Background Color:	Blue
Text Color:	White
Graphics Color:	White
Border:	Double Line
Characters:	Message displayed in sentence format with initial caps centered in window. Pushbutton labels follow pushbutton format.
User Action:	The mouse is used to move the mouse cursor to the pushbutton containing the desired action and depressed.

4.5.1.5 Dialog Windows

The Dialog Window is a rectangular window in which the user enters necessary information in pre-defined fields consisting of pushbuttons, click boxes, check boxes, text entry boxes, labels, and scroll bars. It is illustrated in Figure 4-5.



This will end your action editing session

Figure 4-4: An Example of a Screen with a Message Window.

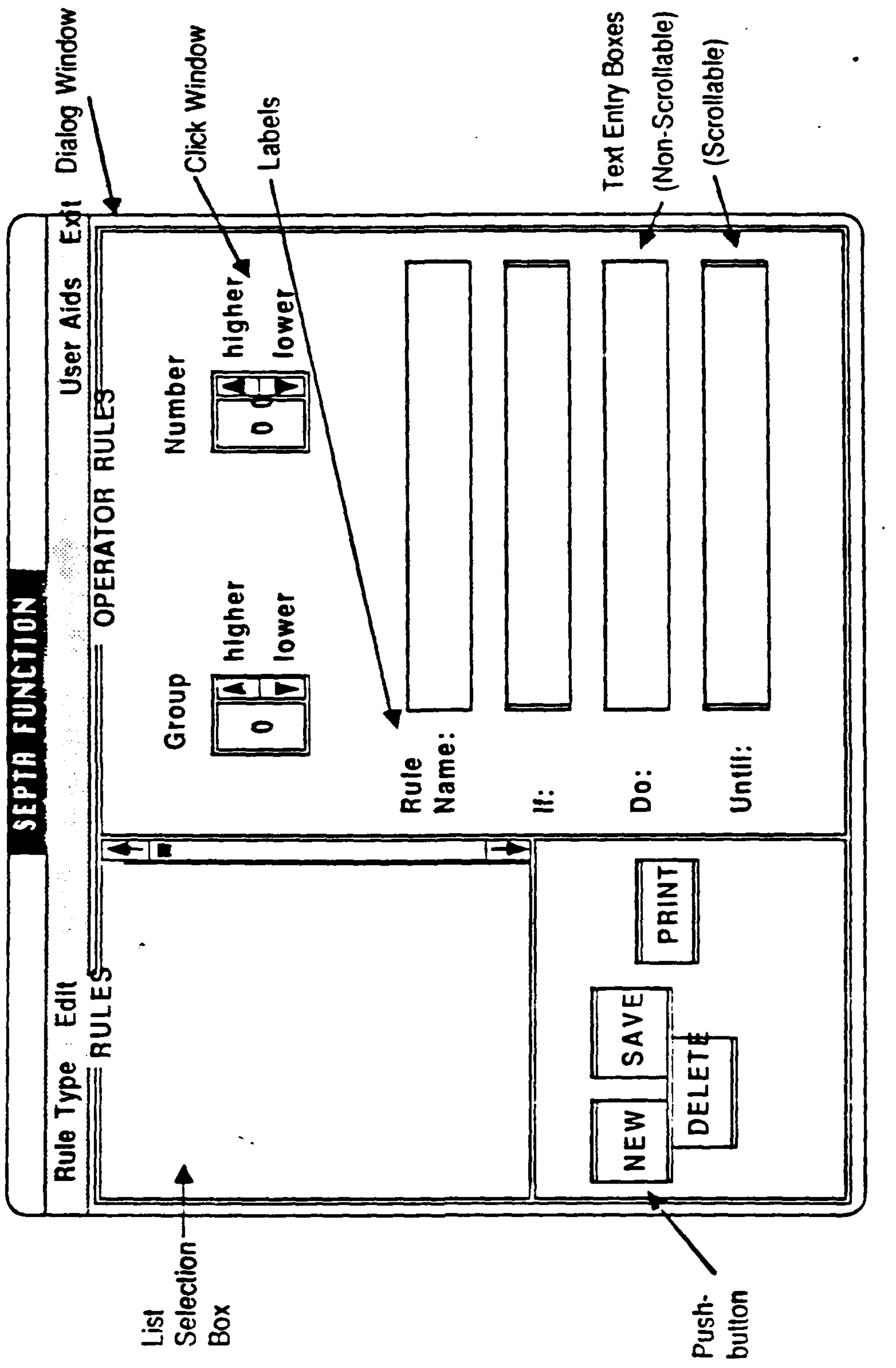


Figure 4-5: An Example Showing a Dialog Window and Other Screen Components

Location: A separate rectangular window that is located in the workspace beneath the menu bar.

Background Color: White

Text Color: Black

Graphics Color: Blue

Foreground Color: Blue

Border: Double line

Characters: Title in upper case centered in top line of window.

User Action: The text cursor (cyan background, black foreground) is initially placed in the beginning of the first text entry box for the user to enter the indicated information. When the entry is completed, the user can depress the return key to move the text cursor to the next item in the sequence or, alternatively, use the mouse to move the mouse pointer to the desired field and click to obtain the text cursor in the desired location. In addition, the tab key can be used to move forward to the next text entry field.

4.5.1.6 Information Windows

Presents informative messages to the user about current system activity and is illustrated in Figure 4-6.

Location: A separate rectangular window that is located in the workspace beneath the menu bar.

Background Color: Blue

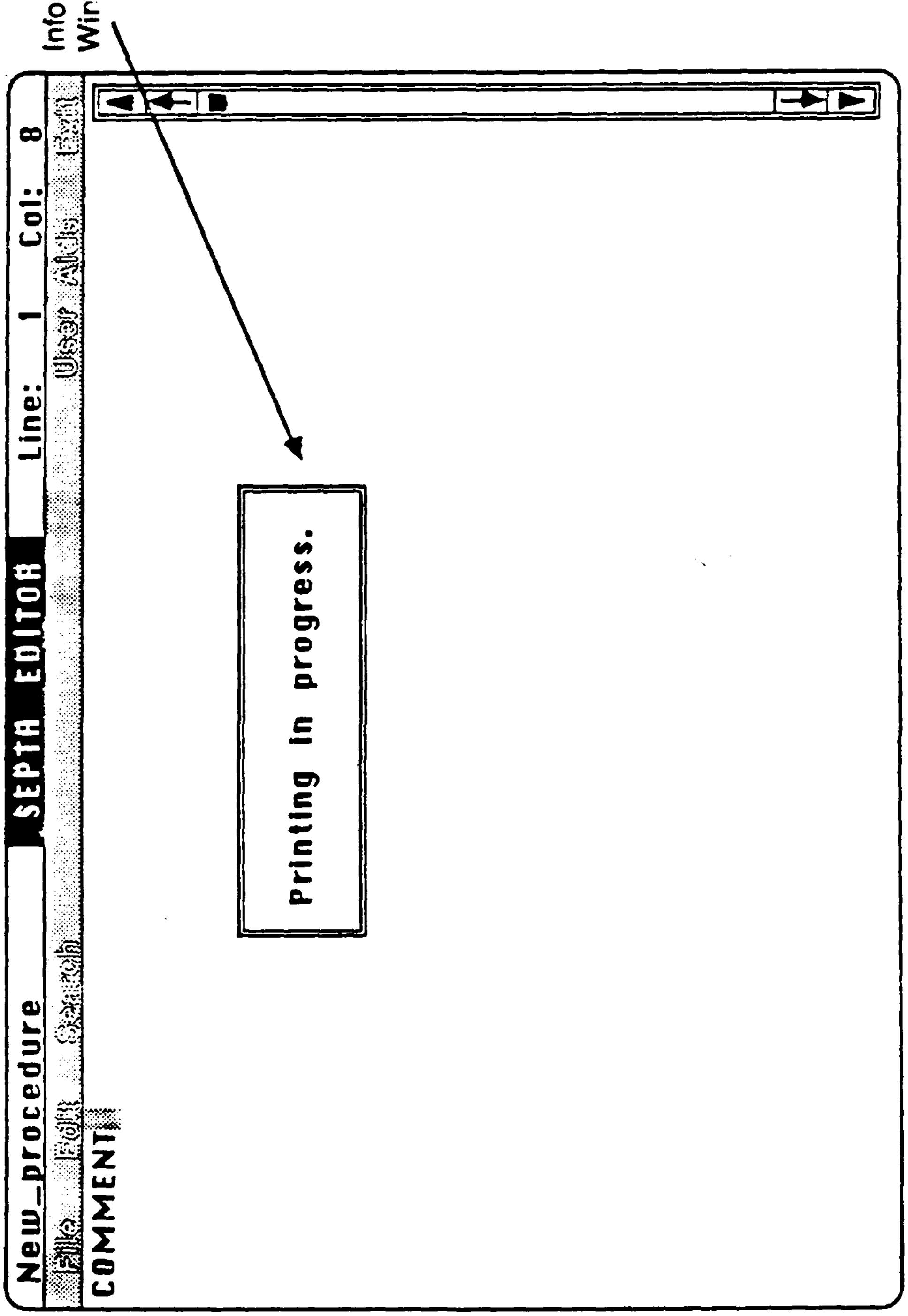
Text Color: White

Graphics Color: White

Border: Double line

Characters: Message displayed in sentence format with initial caps centered in window.

User Action: Information only, no user response permitted in the information window area of the screen.



Printing in progress.

Figure 4-6: An Example of an Information Window.

4.5.1.7 Text Entry Screens

This is a user scrollable window in which the user can enter textual/numerical information in a free-format. A scroll bar is displayed on the right side of the window and is illustrated in Figure 4-3.

Location:	A separate rectangular window that is located in the workspace beneath the menu bar.
Background Color:	White
Text Color:	Black
Graphics Color:	Blue
Border:	Double Line
Characters:	Title in Upper Case; contents dependent upon user entries.
User Action:	The rectangular text cursor is initially placed at the upper left corner of the window. The user can use the mouse or arrow keys (right, left, up, down, home, page up, and page down) to position the text cursor to the location where the next keyboard entry is to be placed. All key strokes are inserted at the current location of the text cursor. If the entry causes the length of the current line to exceed the display width, all text following the previous delimiter (space) is moved to the next line. The depression of a carriage return moves the text cursor and any text after it to the next line. The home key moves the text cursor to first window of text; the end key moves the text cursor to the last window containing text.

4.5.1.8 List Selection Box

A list of all items available to the user for the current function, e.g., list of rules for the rule editor, actions for the action editor, objects for the object editor, etc. It requires a scroll bar on the right side of the window to be used to alter the viewing area of the window. The list box window may obscure the previous contents of the screen while it is active. It is illustrated in Figure 4-5.

Location:	A separate rectangular window that is located in the workspace beneath the menu bar.
Background Color:	White
Text Color:	Black

Graphics Color: Blue

Border: Double Line

Characters: Title in upper case in center of top line of window; pushbutton labels follow pushbutton format; remainder dependent upon user inputs.

User Action: The mouse pointer is initially located on the first item in the window. The mouse is used to move the mouse cursor so as to point to the name of the item to be selected. The currently selected item is shown in reverse video. If the desired item is not currently displayed within the window, the user can move the mouse cursor to the red triangles located at either end of the scroll bar and then depress the mouse button. Each click on the red triangle will display the next set of items in the window. If the user clicks on the up (down) triangle and the pointer is already located at the first (last) item, the contents of the screen will remain identical. Once the desired item is selected, the mouse cursor must be moved to the appropriate pushbutton to invoke the desired action.

4.5.1.9 List Viewing Box

A list of all defined items for the user to view, e.g., list of alphabetics for the object editor, etc. It requires a scroll bar on the right side of the window to be used to alter the viewing area of the window. The list viewing box window may obscure the previous contents of the screen while it is active. It is illustrated in Figure 4-7.

Location: A separate rectangular window that is located in the workspace beneath the menu bar.

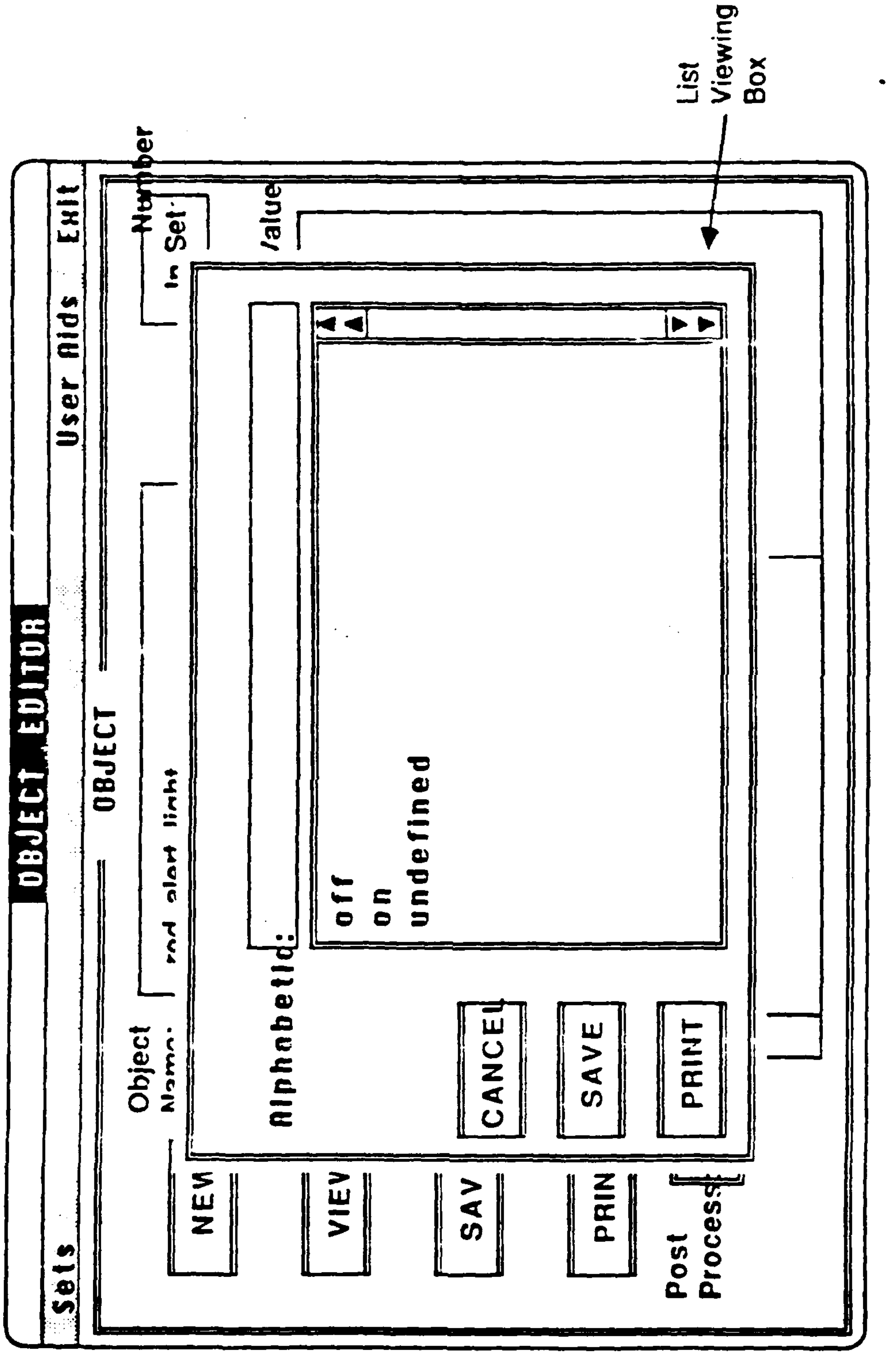
Background Color: White

Text Color: Black

Graphics Color: Blue

Border: Double Line

Characters: Title in upper case in center of top line of window; pushbutton labels follow pushbutton format; remainder dependent upon user inputs.



Object Name: red alert light

Alphabetic:

off
on
undefined

Figure 4-7: An Example of a List Viewing Box.

User Action: The mouse pointer is initially located on the first item in the window. If the desired item is not currently displayed within the window, the user can move the mouse cursor to the red triangles located at either end of the scroll bar and then depress the mouse button. Each click on the red triangle will display the next set of items in the window. If the user clicks on the up (down) triangle and the pointer is already located at the first (last) item, the contents of the screen will remain identical. Once the viewing of the defined items is complete, the mouse cursor must be moved to the appropriate pushbutton to invoke the desired action.

4.5.1.10 Push Buttons

Pushbuttons perform instantaneous action as described by the text label with a mouse click anywhere within the button area. They are illustrated in Figure 4-5.

Location: A separate rectangular window that is located in the workspace beneath the menu bar.

Background Color: Assumes background color of item beneath.

Text Color: Red

Graphics Color: Assumes foreground color of item beneath.

Border: Rectangular box with double line on top and bottom; single line on left and right. It is sized so that there are at least two leading and trailing spaces around the pushbutton legend.

Characters: Upper case button label centered in box.

User Action: The mouse is used to move the mouse cursor so that the pointer is located anywhere within the rectangular area and then a mouse button is clicked. Once any mouse button is depressed, the button label is shown in reverse video (red background and white foreground) and the indicated function immediately invoked.

4.5.1.11 Click Boxes

Click boxes are used to specify numeric values and are illustrated in Figure 4-5. It requires a scroll bar on the right side of the window to be used to alter the numeric value currently displayed in the window.

Location: Varies but always within a dialog window.

Background Color: White

Text Color. Black

Graphics Color: Blue

Border: Rectangle with double line border.

Characters: Title in upper case in center of top line of window.

User Action: The user can modify the currently displayed value by:

- Moving the mouse pointer to the red triangle located above the box to increase the number shown in the box by 1 each time a mouse button is depressed or
- Moving the mouse pointer to the red triangle located beneath the box to decrease the number shown in the box by 1 each time a mouse button is depressed.

4.5.1.12 Text Entry Boxes

Text entry boxes are fields where textual or numerical data are entered and are illustrated in Figure 4-5.

Location: Varies but always within a dialog window.

Background Color: White

Text Color: Black

Graphics Color: Blue

Border: Rectangular box drawn with single blue line. If the entry in the box can be bigger than the box size, a double blue line is placed on the left and right to indicate that the user can scroll right and left within this box.

Characters: Label with initial caps terminated with a colon to the left of the box; entries in box are based upon user actions.

User Action: The rectangular text cursor is initially placed at the left side of the box. The user can use the mouse or arrow keys (right and left) to position the text cursor to the location where the next keyboard entry is to be placed. All keyboard strokes are inserted at the current location of the text cursor. If the entry causes the length of the current line to exceed the display width, either of the following will occur:

- If the box is the exact size of the permitted entry (i.e., the nght and left side are single lines), a beep will be sounded and future keyboard entries (except backspace and delete) will be ignored until a non-text key is depressed or
- If the entry can be larger than the box (i.e., the nght and left sides are double lines), the text will be scrolled to the left as additional keys are depressed until the maximum field size is reached

The left and right arrow keys move the cursor one space in the indicated direction within the text entry box. The home key moves the text cursor to the first character in the box; the end key moves the text cursor to the last character in the box.

4.5.1.13 Scroll Bar

Scroll bars are used to change which part of a list of items (list window) or contents of a file (text entry window) is shown the window. Double red scroll arrows are used at the top and bottom of the scroll bar rectangle to indicate the direction the viewing area is to be moved. The top arrow (s) is used to scroll up one line at a time; the down arrow (t) is used to scroll down one line at a time. The second up arrow (≠) is used to scroll up one page at a time; likewise the top down arrow (O) is used to scroll down one page at a time. The scroll bar is illustrated in Figure 4-3.

Location:	Rectangular box shown on right side of text entry and list boxes.
Background Color:	White
Text Color:	None
Graphics Color:	Blue
Border:	Double line.
Cnaracters:	Graphics characters of t and s.
User Action:	The user uses the mouse to position the mouse cursor at the desired scroll arrow and clicks to alter the contents of the window. The content of the window is moved in the opposite direction from the arrow. For example, when the user clicks the top scroll arrow, the contents moves down, bringing the view closer to the top of the list or document. Each click of the single arrow moves the window contents one line in the chosen direction; each click of the double arrow moves the window contents one page in the

chosen direction. Continuous depression of the mouse results in continuous movement in the chosen direction. Once the top or bottom of the window contents is reached, depression of the scroll arrows in that direction are ignored.

4.5.1.14 Labels

A label is an alphanumeric description of the information to be entered for a component of a dialogue window and is illustrated in Figure 4-5.

Location:	Varies
Background Color:	White
Text Color:	Blue
Graphics Color:	None
Border:	None
Characters:	Initial upper case and terminated with a colon.
User Action:	None

4.5.1.15 Mouse Cursor

The mouse cursor is a pointing device used to select commands from menus, to control cursor movement, and to manage file scrolling. It is illustrated in Figure 4-3.

Location:	Varies
Background Color:	Assumes background color of object beneath.
Text Color:	None
Graphics Color:	Assumes foreground color of object beneath.
Border:	None
Characters:	Arrow (↖)
User Action:	The user moves the mouse cursor to the desired location by moving the mouse in the desired direction. Every mouse movement moves the mouse cursor in exactly the same way. The following mouse actions can be performed: • **Clicking** — positioning the mouse cursor with the mouse, briefly pressing and releasing the mouse button without moving the mouse.

- **Pressing** — positioning the mouse cursor with the mouse, holding down the mouse button without moving the mouse.
- **Dragging** — positioning the mouse cursor with the mouse, holding down the mouse button, moving the mouse to a new position. then releasing the button

4.5.1.16 Text Cursor

The text cursor indicates where next keyboard stroke will be entered and is illustrated in Figure 4-3.

Location:	Varies
Background Color:	Cyan
Text Color:	Black
Graphics Color:	Assumes foreground color of object beneath.
Border:	None
Characters:	Rectangle the size of a single character (†).
User Action:	The user can use the arrow keys or the mouse to indicate where the text cursor should be positioned. Each subsequent keyboard stroke will be inserted at the location of the text cursor.

4.5.1.17 Text Viewing Window

A text viewing window is a user scrollable window in which the user can view the content of a file. A scroll bar is displayed on the right side of the window and is illustrated in Figure 4-8.

Location:	A separate rectangular window that is located in the workspace beneath the menu bar.
Background Color:	White
Text Color:	Black
Graphics Color:	Blue
Border:	Double Line
Characters:	Title in Upper Case; contents dependent upon user entries.
User Action:	The rectangular text cursor is initially placed at the upper left corner of the window. The user can use the mouse or arrow keys (right, left, up, down, home, page up, and page down) to alter the current contents of the window.

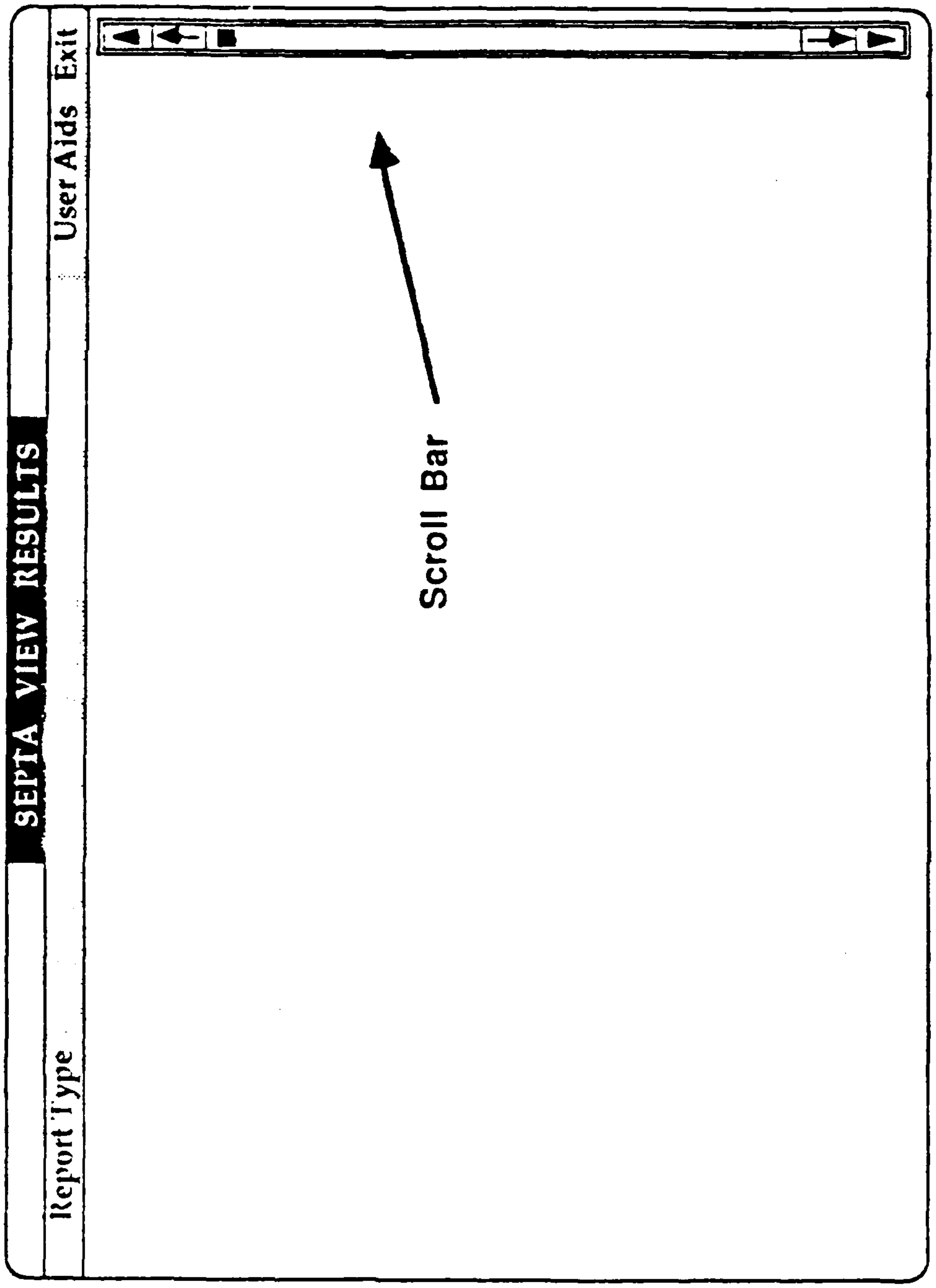


Figure 4-8: Screen Showing Text Viewing Window.

4.5.1.18 Check Boxes

Location:	Varies but always within a dialog window.
Background Color:	White
Text Color:	Black
Graphics Color	Blue
Border:	Rectangle enclosed with double lines
Characters:	Check box label to the right with initial caps
Description:	Check boxes are used as toggle switches for information that may assume either an on or off value. The value is on if the box contains an X; otherwise it is off and contains a blank.
User Action:	Use the mouse to move the mouse cursor so that the pointer is located anywhere within the rectangular check box area and click. Once any mouse button is depressed, the value of the check box will be toggled with the X disappearing if it was previously checked and appearing if it was previously not checked. Each click of the mouse button while the mouse cursor is within the check box will result in a change of value.

4.6 Input Devices

4.6.1 Keyboard

The UCI input devices consist of two complementary devices — a keyboard and a mouse. The keyboard is used to enter alphanumeric data and as an alternative to the use of the mouse to move the text cursor between input fields on dialogue boxes. The keyboard consists of a standard typewriter keyboard, numeric keypad overlaid with cursor move keypad, and a set of function keys as discussed later. The overlay of the numeric and cursor keypads is one reason why most people prefer a mouse to cursor keys.

The numeric keypad includes keys for the numbers zero through nine arranged in an adding machine format; it also has keys for special functions (such as a minus sign, equal sign, etc.) and is used to speed entry of numeric information. The cursor movement keypad contains an up-arrow, down-arrow, right-arrow, left-arrow, home, and end keys and are used to control cursor movement within a window as an alternative to the use of the mouse. The user controls the functioning of the keypad as either a numeric pad or cursor movement pad through the use of the **NUM LOCK** key. Above the num lock key is a small red dot light. If the light is lit, then the keypad is functioning as a numeric pad; otherwise it functions as a cursor movement pad. The special function keys are used to select menu options as an alternative to the use of the mouse for experienced users with certain modules.

The following keys have special functions as described within the indicated SEPTA modules:

- **Enter (ø)** is used for the following:

  Move the text cursor and any subsequent text to the next line.

  Move the text cursor to the next data entry field in the dialog window.

- **Backspace (˜)** deletes the character to the left of the text cursor. If the text cursor is positioned at the top, leftmost character in a window, subsequent depressions of the backspace key are ignored.

- **Tab (|˜)** is used for the following:

  Insert up to five blank characters in the text depending upon the current cursor position. If the entry of the blank characters causes the width of the line to exceed the screen width, the text cursor and any subsequent text will be moved to the next line.

  Move the text cursor to the next input field in the dialog window.

- ~/ʳ key is used to generate the underscore character __ regardless of the status of the shift and/or shift lock keys.

The following keys on the cursor movement pad have special functions as described for the indicated SEPTA modules:

- **HOME** (above 7 key on numeric pad) is used for the following:

  Move the text cursor to the leftmost character in the first entry field in the dialog window,

  Move the text cursor to first screen containing text and maintain the relative position of the text cursor, and

  Move the text cursor to the leftmost character in a text entry box

- **Up-arrow** (≠)— (above 8 key on numeric pad) moves the text cursor up one line. If the current line is the top line on the page, the depression of the up arrow scrolls the page. If the current line is the first line then subsequent depressions of the up-arrow are ignored.

- **PgUP** (above 9 key on numeric pad) moves the text cursor to the previous page of text and maintains the relative position of the text cursor on the page. The top line of the previous page becomes the bottom line of the current page.

- **Right-arrow (Æ)** (above 6 key on numeric pad) moves the text cursor one character to the right. (All modules).

- **Left-arrow** (¯) (above 4 key on numeric pad) moves the text cursor one character to the left.
- **END** (above 1 key on numeric pad) is used for the following:

  Move the text cursor to the rightmost character in the last entry field in the dialog window,

  Move the text cursor to last screen containing text and maintain the relative position of the cursor, and

  Move the text cursor to the rightmost character in a text entry box.
- **PgDN** (above 3 key on numeric pad) moves the text cursor to the next page of text and maintains the relative position of the text cursor on the page. The bottom line of the previous page becomes the top line of the current page.
- **Down-arrow** (Ø)— (above 2 key on numeric pad) moves the text cursor down one line. If the current line is the last line of the page, the down-arrow will scroll the text down one line. If the current line is the last text line, subsequent depressions of the down-arrow will be ignored.

The following functions keys are used for the following within the Editors:

- **F1** — Begin text marking for cut/copy operation.
- **F2** — End text marking for cut/copy operation.
- **F3** — Cut text.
- **F4** — Copy text.
- **F5** — Paste text.
- **F6** — Clear text.

4.6.2 Mouse

The mouse is used as a pointing device to select commands from menus, to control cursor (pointer) movement, and to manage file scrolling. In SEPTA, the standard pointer is an arrow (/). Every move you make with the mouse moves the pointer in exactly the same way. The following terms describe various actions associated with mouse utilization:

- **Clicking** — positioning the pointer with the mouse, briefly pressing and releasing the mouse button without moving the mouse.
- **Pressing** — positioning the pointer with the mouse, holding down the mouse button without moving the mouse.
- **Dragging** — positioning the pointer with the mouse, holding down the mouse button, moving the mouse to a new position, then releasing the button.

5. SETUP PROGRAM DESCRIPTIONS

In this section, we describe the program modules in detail to set up the simulations. The user interface was described in detail in the previous section. Accordingly, in this section we simply show the screens for a module and list the options for each screen. Some programs, particularly subroutines, are used in various places. These are referenced in this section and will be discussed in detail in Section 9.

5.1 System Performance Criteria and Job Correspondence

Two pieces or sets of information SEPTA will ultimately need come from products MMP1 and MMP5. Fortunately, a common data language has been agreed upon. The taxonomy to be used in the MMP product set is that of Kaplan and Crooks (1980), according to contractor agreement. The general outline and structure of this taxonomy is shown in Table 5-1. SEPTA uses input from both MMP1 and MMP5 based on this taxonomy. Since it is anticipated that different individuals (and organizations) will be using MMP1 and MMP5, it is quite possible for changes to be made in the taxonomy, or for it to be used in somewhat different ways in the two products. Accordingly, the MMP6 user needs to be advised in the event there are differences between the two data sets. These differences could take a variety of forms such as missing task descriptions in one of the files, non-matching labels due to misspellings, etc. This program module will check to see if the jobs as specified from MMP5 meet the system design performance requirement and match up with the from MMP1. This is an absolute requirement or we will not be able to link the information from MMP1 with a particular maintainer or operator.

For example, in the preliminary taxonomies we have received, there are some discrepancies. Under Operational functions for rifles **Detect / Locate Targets** includes a) search for target, b) detect/locate target, c) identify friend or foe. Under Operational function for infantry fighting vehicles, **Acquire Target** involves the identical breakdown, a) search for target, b) detect/locate target, c) identify friend or foe. While the detailed behavioral / cognitive components of "search for target" will be the same, or at least very similar, SEPTA will not recognize the different labels. An example of the operational function taxonomy is illustrated in Table 5-2. Further, several contractors have indicated the necessity for planned divergence from the taxonomy. Clearly, there is need for a comparison module.

Table 5-1: Listing of Structural Components of Kaplan and Crooks (1980) Taxonomy.

Goals:

Air superiority

Missions:

Destroy aircraft

Largest, measurable, unitary component

System Functions (both man and machine)

Maneuver to attack position, deliver the ammunition on or near the target

Tasks:

Operation:

Detect targets, orient weapon system, illuminate/designate target

Maintenance:

These tasks are to be stated in the most general, measurable terms i.e., "at a sufficiently general level of detail."

Conditions:

Influences on system performance

Weather, terrain, target, personnel, training, tactics

Figure 5-1 shows the concepts in matching data coming from Product 1 and Product 5. While both MMP1 and MMP5 use the structure of the Kaplan and Crooks (1980) taxonomy and therefore a common structure reduces a potentially difficult problem, the communality will not be 100%. Accordingly the program will need to insure the appropriate format for SEPTA, insuring that each of the functions and tasks in MMP1 has a counterpart in MMP5, and assigning each task in MMP1 a job according to MMP5. This needs to be checked in only one direction. If for some reason, a mission does not require a task to be performed, there is no need to have that task assigned to any individual. MMP1 uses separate files for function and task while MMP5-S uses a common hierarchical file, ignoring mission.

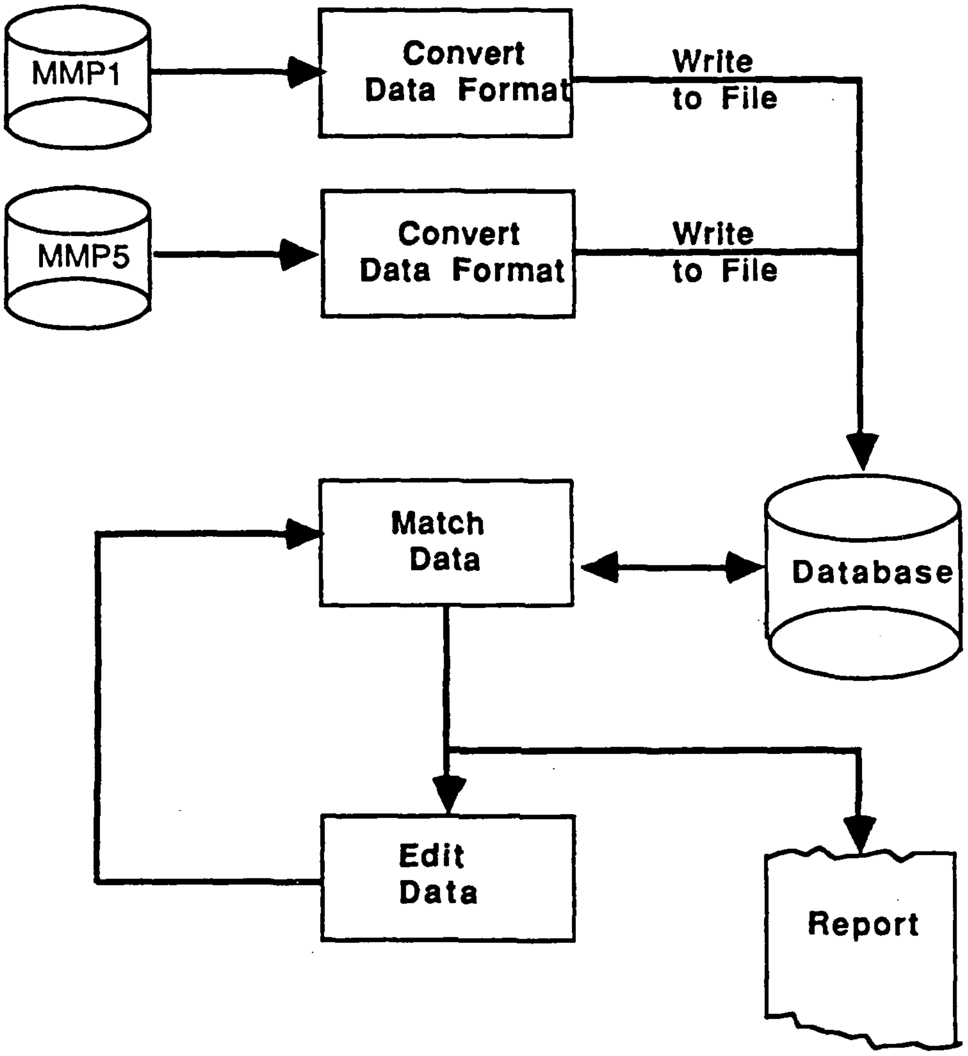


Figure 5-1: Illustration of the Flow in Matching Data Coming from Product 1 and Product 5.

The program will have several things to do. First, there is the compatibility of files; we will reorganize the multiple files for a single mission from MMP1 into a single file and create SEPTA formats for both MMP1 and MMP5. Second, the program will cross-check between MMP1 and MMP5 to insure common data elements. At this point a discrepancy report will be generated, if necessary, and editing can take place to correct and eliminate discrepancies. Third, the program will assign personnel (jobs) to tasks in MMP1, using the information in MMP5.

Table 5-2: Sample of Operational Function Taxonomy

The following is taken from material provided the MMP1 contract team. This particular excerpt focuses on operation; similar taxonomies are available for maintainers. (This represents preliminary material still in preparation and, therefore, subject to change.)

Operation Functions for Tanks

PLAN AND PREPARE MISSION

- Receive/Review Order
- Adjust/Boresight Weapon Systems
- Adjust/Inspect Other weapon systems
- Enter Data onto Onboard Computer(s)
- Prepare Vehicle/Personnel for NBC Environment

DRIVE VEHICLE

- Start engine
- Check Controls/Instruments
- Drive Vehicle (non-tactical movement)
- Perform Tactical Movement
- Perform Water Crossing

NAVIGATE

- Identify Present Location
- Identify Destination
- Select Travel Route
- Estimate Time of Arrival and Fuel Requirements

COMMUNICATE

- Transmit/Receive Messages
- Encode/Decode Messages
- Communicate Using Countermeasure Procedures

ACQUIRE TARGET

- Search for Target
- Detect/Locate Target
- Identify Friend or Foe

ATTACK TARGET

- Attack Target
- Move into Firing Position
- Select Target(s)

- Select Weapon(s) and Ammo
- Aim/Sight Weapon
- Track Target
- Fire Weapon
- Clear/Unload Weapon
- Sense Round/Adjust Fire
- Leave Firing Position

DEFEND AGAINST ATTACK

- Deploy to Cover
- Identify/Locate Source of Enemy Fire
- Perform Evasive Maneuvers
- Employ ECCM
- Dispense/Disperse Smoke

DEFEND/OCCUPY TERRAIN

- Select Position
- Camouflage Vehicle
- Select Reference Points
- Develop Range Cards
- Leave Position

PERFORM RECONNAISSANCE

- Move to Reconnaissance Area
- Obtain Tactical Information
- Transmit Tactical Reports

CALL FOR DIRECT SUPPORT

- Call for/Adjust Indirect Fire
- Call for/Adjust Illumination or Smoke

PERFORM POST-MISSION TASKS

- Shut Down Engine
- Power Down Other Systems
- Perform

COMPENSATE FOR EQUIPMENT MALFUNCTIONS AND EMERGENCIES

- Identify Malfunction
- Identify Source of Malfunction
- Compensate for/Recover from Malfunction
- Evacuate Vehicle
- Extinguish Fire
- Clear Misfire on Small Arms

5.1.2 Screen: Menu Options.

The main screen is shown in Figure 5-2 and the various options for the conversion process are as follows:

- **User Aids** — provides the capabilities to view help files.
- **Exit** — terminates the conversion and returns the user to the MMP6 main menu.
- **Convert** – convert incoming data to R:BASE.

  Name files – Assign names to new files according to system profile.

  Select missions - from MMP1 and the System Missions file.

  MMP5 conversion to R:BASE.
- **Match data** – run comparison between converted MMP1 and MMP5 files.
- **Edit File** – modify new MMP1 and/or MMP5 files. This is a general editor based on R:BASE routines. See Section 9 for details.
- **Select File** – Select an MMP1 file or MMP5 file for later simulation use.
- **Report Generator** - his is a general routine based on R:BASE routines. See Section 9 for details.

  Exception – Comparison report giving exceptions from Discrepancy file

  Contents – Print contents of converted files

5.1.3. File Input

- MMP1 files. DIF format file. See Appendix B for detailed record descriptions for MMP1 files.
- MMP5 file. DIF format file. See Appendix C for detailed record descriptions for MMP5-S. (MMP5-M will be similar to the record layout for MMP1.)

The data files created will use the field descriptions and the field lengths in Appendixes B and C. These are preliminary design specifications from MMP1 and MMP5-S and are likely to change as those designs mature. While the details will undoubtedly change, the basic structure of the data files is probably set as the requirement of checking the files for consistency.

5.1.4. File Output

- To Report Generator (Discrepancy file).
- From original MMP1 and MMP5 File contents into SEPTA format
- File to be used in dynamic simulation. Either MMP1 reconstructed file for a mission or MMP5.

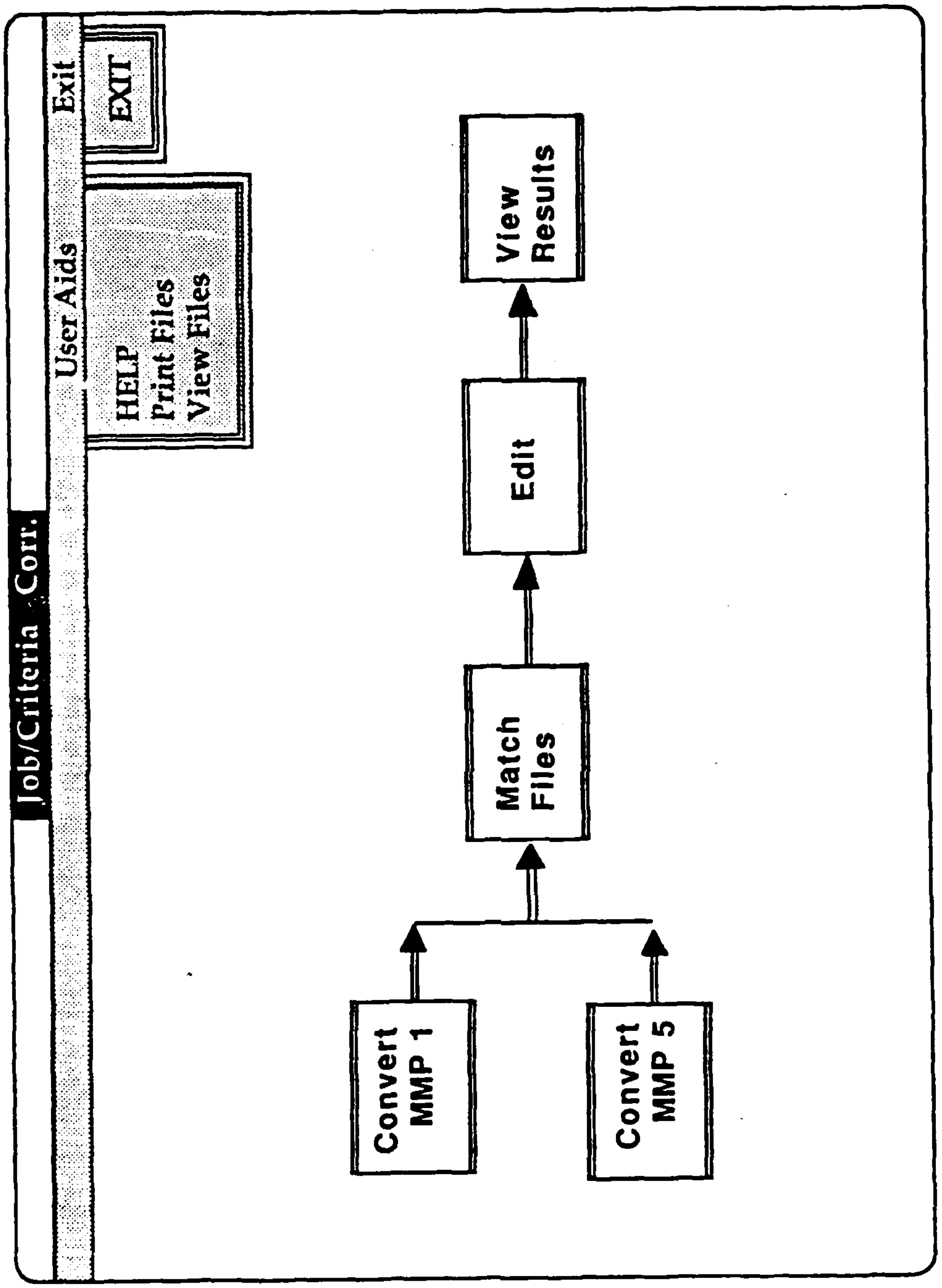


Figure 5-2: Job / Criteria Correspondence Screen

5.2 Hypothetical Individual Generation (HIG)

"Hypothetical individuals" refers to the concept of an individual being randomly selected from a sample (or population) description. Both the dynamic and static simulations employ this concept. The sampling technique is one used in CAR (Section 7) and some validation results from this procedure are discussed in that section. The data requirements are the means and standard deviations. If the data are skewed (as reaction time often are) it is helpful to have maxima and minima as well to insure the sampled observation remains in bounds. Because the variables (means) are correlated, coming from multivariate distributions and not independent distributions, the sampled values have to be adjusted to take the correlations into account. Otherwise a set of unlikely and unrealistic individuals would be created.

The principles (and subroutines) used to generate the hypothetical individuals for dynamic and static simulations are identical. The point of divergence is in the data sets used, anthropometric and personnel characteristic data (mainly Project A). To accomplish this, the appropriate identifiers are passed from the calling program to the subroutines so the appropriate data are used. These identifiers are the size of the vectors and matrix and the data file name.

The files containing the hypothetical individuals are maintained on disk. Accordingly, this module need only be exercised when the user wishes to start a simulation using an hypothetical individual other than the "average" which is the default subject.

Because the user may (and should) want to enter a hypothetical individual of interest other than the average, options are provided for this possibility. The Hypothetical Individual Generation module sets up the dynamic simulation by providing a subject description based on the known personnel characteristic observations. There are two different paths which the user will be able to follow to enter or create hypothetical individuals depending on his current purpose. One path will generate the description of an individual to be tested based on means, that is, a hypothetical average individual. The other path can be used to input a description (specific personnel measurement values) of the individual to be tested. Additionally, the user will be able to speciry whether he wishes to work on personnel performance characteristics or anthropometric characteristics.

For the dynamic simulation, the approach will be to use the mean (average) personnel values in the simulation or allow the user to enter a specific set of personnel values and then adjust these values appropriately. In the latter case, the program will adjust the table of KUIP values to correspond to the Hypothetical Individual. The KUIP values will be used in the dynamic simulation as numeric values for simulation times and accuracies of the various micromodels. When the simulation is complete, the other individuals will be used in the ANALYSIS module described later in Section 8.

The Hypothetical Individual Generator (HIG) module of SEPTA maintains all information for soldier samples and generates the soldier data that will be required by STATSIM and DYNSIM. HIG can be used to generate the following type of soldier data:

- Perceptual, cognitive, and motor performance measurements based upon Product A data.
- Anthropometric (or body) measurements based upon standard anthropometric measures such as stature, waist height, sitting-eye height, etc.

When defining the characteristics of the soldier sample, the parameters for individual soldiers can be input directly into HIG or HIG can generate a sample of soldiers from population statistics. If the user desires the latter option, HIG requires input of the means, standard deviations, and an intercorrelation matrix of a set of measurements. Based upon the Monte Carlo technique used by CAR, HIG constructs a sample of soldiers whose measures reflect those of the population.

HIG permits the user to enter data in either inches or centimeters and the HIG-user dialogue will request that the user specify the desired units. The data must be either all in inches or all in centimeters and the units cannot be mixed. All output routines also prompt the user to determine what units should be used in the output reports. This permits the user to enter data in centimeters but obtain analysis reports in either inches or centimeters. Within HIG, all measurement data will be stored in inches. HIG will convert metric units to English units for internal storage and processing purposes.

STATSIM and DYNSIM use the tables created by HIG to define the individual(s) that will be analyzed in the simulation. HIG must be only rerun when new population statistics are obtained or a different individual is to be simulated.

5.2.1 Description

HIG maintains the tables related to the defined sample measurements and permits the user to view, edit, delete, and create soldier samples. The basic data required for each sample includes the following types of information:

- Description: descriptive information about the sample including sample name, sample type, gender, sample size, number of measurements, etc.
- Means and Standard Deviations: measurement identifier, mean, standard deviation, and validity checking information for each measurement.
- Correlation Matrix: correlations for each measurement.
- Sample Set: actual or generated measurements for each soldier in the sample.

The correlation matrix is specified as a positive definite matrix that is symmetric about the diagonal. Therefore, only the upper triangular portion need be entered. This format for the correlation matrix is illustrated below:

Table 5-3: Illustration of Correlation Matrix.

	1	2	3	4	...	...	...				N
1	-	r_{12}	r_{13}	r_{14}							r_{1i}
2	-	-	r_{23}	r_{24}							r_{2i}
3	-	-	-	r_{34}							r_{3i}
.											
.											
.											
J-1	-	-	-	-	-	-	-	-	-	-	r_{j-1i}

where:

N equals the number of measurements,

i equals the measurement row one through N,

j equals the measurement column one through N, and

R_{ij} equals the correlation between two measurements.

HIG is organized into the following modules:

HIGMAIN: high-level system controller that manages the lower-level functions and interacts with the user to invoke the lower-level functions.

BODYSAMP: maintains the tables associated with the body samples including creation of new samples, modification of existing samples, and deletion of existing samples.

PERFSAMP: maintains the tables associated with the performance samples including creation of new samples, modification of existing samples, and deletion of existing samples.

GENSAMP: generates samples of body and/or performance samples and stores the result in the sample tables.

The organization of the HIG is presented in Figure 5-3 with the left to right progression showing the sequence in which the modules must be invoked in order to properly construct , generate, and view the soldier sample.

5.2.2 HIGMAIN

5.2.2.1 Description

HIGMAIN is the main module in HIG and is invoked when HIG is invoked. It is the top level module that controls interactions with all other modules of the HIG system. There are four commands which can be executed from the main screen of HIG — Define Body Sample, Define Performance Sample, Generate Sample, and View Sample. A pushbutton was created for each function and the user invokes the desired function by moving the mouse over the title of the selected pushbutton and clicking the mouse. Whenever the user terminates one of the lower level modules, control is returned to HIGMAIN for the selection of the next module or to return to the top-level SEPTA system.

5.2.2.2 HIGMAIN Screens

The HIGMAIN screen contains a diagram of pushbuttons and arrows indicating the functional flow of steps invoked in defining and generating a sample as illustrated in Figure 5-3.

HIGMAIN Title Bar. The title bar of HIGMAIN contains the words 'Hypothetical Individual Generator' as the function name. HIGMAIN does not use the current activity or status information areas of the title bar.

HIGMAIN Menu Bar. HIGMAIN contains the following menu options of the menu bar as illustrated in Figure 5-3:

- **User Aids** — provides the capabilities to view help files.
- **Exit** — terminates HIGMAIN and returns the user to the MMP6 main menu.

The **User Aids** pull-down menu contains the following commands:

- **Help** command that allow the user to obtain help windows about the HIGMAIN module.
- **Print Tables** command that allows the user to obtain a copy of any of the HIG tables on the printer.
- **View Tables** command that allows the user to obtain a copy of any of the HIG tables on the screen.

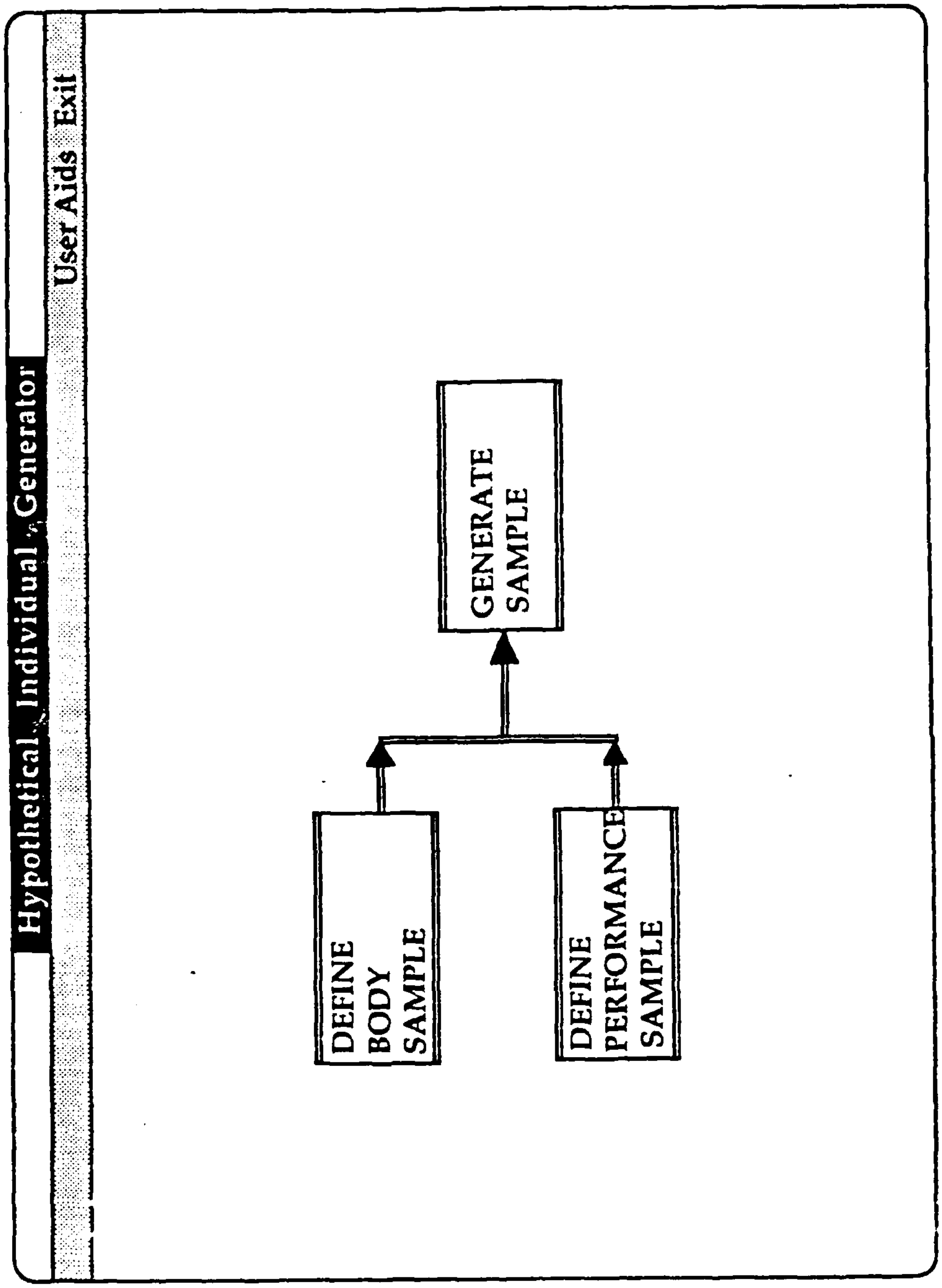


Figure 5-3: HIGMAIN Screen.

HIGMAIN Main Screen. The HIGMAIN main screen contains the following pushbuttons:

- **Define Body Sample** — calls the BODYSAMP module to allow the user to define the necessary parameters to generate a sample of body measurements.
- **Define Performance Sample** — calls the PERFSAMP module to allow the user to define the necessary parameters to generate a sample of performance measurements.
- **Generate Sample** — calls the GENSAMP module to generate a body and/or performance sample.

5.2.3 BODYSAMP

5.2.3.1 Description

The BODYSAMP module of HIG maintains information for all user-defined samples of anthropometric or body measurements. When defining soldier characteristics, the body measures of individual soldiers can be input directly into HIG or HIG can generate a sample of soldiers from population statistics. If the user desires the latter option, HIG requires input of the means, standard deviations, and an intercorrelation matrix of the set of body measures. The population statistical data required by HIG for body measurements is readily available from anthropometric source books for a variety of populations — gender, nationality, age, occupation, etc. The basic source book for MMP6 is the NASA Source Book (NASA, 1978). In order to clearly identify the anthropometric measurements, the numeric code contained in the NASA Source Book will be used to identify measurements, e.g., the Stature measurement would be identified as numeric code 805. It is also assumed that the measurements are obtained using the description contained in the NASA Source Book. The current set of anthropometric measurements required in order to transform external body measurements into the internal-link person structure required by STATSIM are the following (the numbers in parenthesis are the NASA measurement identifier numbers):

- **Stature** (805) — vertical distance from the standing surface to the top of the head.
- **Waist Height** (949) — vertical distance from the standing surface to the waist landmark.
- **Sitting Height** (758) — vertical distance from sitting surface to the top of the head.
- **Eye Height, Sitting** (330) — vertical distance from the sitting surface to the outer corner of the eye.
- **Popliteal Height, Sitting** (678) — vertical distance from the floor to the underside of the thigh immediately behind the knee.

- **Buttock-Knee Length** (194) — horizontal distance from the most posterior aspect of the right buttock to the most anterior aspect of the right kneecap.
- **Shoulder-Elbow Length** (751) — distance from the top of the acromial process to the bottom of the elbow.
- **Forearm-Hand Length** (381) — distance from the top of the elbow to the tip of the longest finger.
- **Bideltold Diameter** (122) — horizontal distance across the body at the level of the deltoid landmarks.
- **Hip Breadth** (457) — maximum horizontal distance across the hips.
- **Foot Length** (362) — distance parallel to the long axis of the foot, from the back of the heel to the top of the most protruding toe.
- **Hand Length** (420) — distance from the wrist landmark to the tip of the middle finger.
- **Thumbtip Reach** (867) — distance from the wall to the tip of the thumb measured with the subject's shoulders against the wall, with the arm extended forward and the index finger touching the tip of the thumb.
- **Buttock-Leg Length** (191) — distance from the base of the heel to a wall against which the subject sits erect with the leg maximally extended forward along the sitting surface.

Pictorial representation of these measurements can be found in the NASA Reference Publication (NASA, 1978).

5.2.3.2 BODYSAMP Tables

BODYSAMP maintains the database tables containing the sample description, means and standard deviations, intercorrelation matrices, sample measurements, and standard measurements descriptions. The elements of the database tables described below contain the RBASE V field types as text, real, note, or integer. The Sample Description table (SAMPDESC) contains the following information:

- Sample Name: A text name containing a maximum of 28 characters. This name is used to link all tables about a particular sample together and appears in all list selection boxes that list the defined samples.
- Sample Description: A text name containing a maximum of 80 characters.
- Sample Type: A 23 character text description containing either the word 'Body' or 'Performance' to indicate the type of soldier data. In this case, the sample type would be Body.
- Gender: A 6 character text description containing either the word 'Male' or 'Female' to indicate the gender of the sample.
- Sample Comments: A note field containing any general comments about the particular sample.

- Number of Measurements: An integer number indicating the number of measurements included in the sample with a maximum of 99.
- Units Type: A 12 character text description containing either the word 'inches' or 'centimeters' to indicate the units of measurements for all data related to the sample.

The Means and Standard Deviation table (SAMPSTAT) contains the following information:

- Sample Name: A text name containing a maximum of 28 characters. This name is used to link all tables about a particular sample together and appears in all list selection boxes that list the defined samples.
- Number of Measurements: An integer number indicating the number of measurements included in the sample with a maximum of 99.

The following elements will be repeated for each measurement:

- Measurement Identifier: An integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- Measurement Description: An 40 character text description that describes the measurement.
- Mean: A real number that contains the means for the measurement.
- Standard Deviation: A real number that contains the standard deviation for the measurement.
- Minimum Value: A real number that contains the lowest acceptable value for the measurement. If not entered by the user, it will be the mean minus 3 standard deviations.
- Maximum Value: A real number that contains the highest acceptable value for the measurement. If not entered by the user, it will be the mean plus 3 standard deviations.

The Intercorrelation table (SAMPCORR) contains the following information:

- Sample Name: A text name containing a maximum of 28 characters. This name is used to link all tables about a particular sample together and appears in all list selection boxes that list the defined samples.
- Number of Measurements: An integer number indicating the number of measurements included in the sample with a maximum of 99.

The following elements will be repeated for each measurement:

- Measurement Identifier: An integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- Measurement Description: An 40 character text description that describes the measurement.
- Correlations: A set of real numbers that contains the correlation values between 0.0 and 1.0. The size of the set is equal to the number of

measurements and each entry is the set must be in the same order as the measurements are organized in the table rows.

The Sample table (SAMPLE) contains the following information:

- *Sample Name: A text name containing* a maximum of 28 characters. This name is used to link all tables about a particular sample together and appears in all list selection boxes that list the defined samples.
- Number of Measurements: An integer number indicating the number of measurements including in the sample with a maximum of 99.
- Sample Size: An integer number indicating the number of soldiers in the sample with a maximum of 400.

The following elements will be repeated for each soldier in the sample:

- Soldier Identifier: A 8 character text field containing a unique identifier for the soldier. If no identifier is entered, it will be automatically generated by HIG and contain a sequential number.
- Measurement Identifier: An integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- Measurement Name: A 40 character text description that describes the measurement.
- Measurement: A real number that contains the value of the measurement for the soldier.

The Standard Measurements *Description table* (MEASDESC) *contains the following* elements for each standard measurement:

- Measurement Identifier: An integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- Measurement Name: A 40 character text description that names the measurement.
- Measurement Description: A 200 character text description that describes the measurement.

5.2.3.3 BODYSAMP Screens

The BODYSAMP screen initially contains the title and menu bars. As the user selects various menu options, the window portion of the BODYSAMP screen will contain the appropriate dialog windows for the user to enter the indicated information.

BODYSAMP Title Bar. The title bar of BODYSAMP contains the words 'Hypothetical Individual Generator' as the function name. The words 'Define Body Sample' will be inserted in the current activity area of the title bar. The status information areas of the title bar will contain various information depending upon the current activity.

BODYSAMP Menu Bar. BODYSAMP contains the following menu options of the menu bar as illustrated in Figure 5-4:

- **Sample Options** — table related commands such as new table, edit table, save table, delete table, etc.
- **Data Type** — indicates the type of data to be entered or viewed as either sample description, means and standard deviations, intercorrelations, or actual soldier measurements.
- **User Aids** — provides the capabilities to print sample information or obtain help messages.
- **Exit** — terminates BODYSAMP and returns the user to the HIGMAIN main menu.

The **Sample Options** pull-down menu contains commands that allow the user to create, save, and define new samples. It contains the following commands:

- **New Sample** — defines a new sample. It closes the current sample after asking if the changes should be saved and then creates a new empty sample. It clears the current dialogue window and puts the description dialogue window on the screen with the text cursor placed in the first character in the sample name field.
- **Open Sample** — opens the tables associated with an existing sample. It closes the current sample after asking if changes should be saved and then opens the sample file selection dialog box. The user views the currently defined sample names and then selects one. The screen is cleared and the sample description dialogue window is placed on the screen with the contents of the description table placed in the appropriate fields. The text cursor is placed in the first character in the sample name field.
- **Save Sample** — saves the currently defined sample. First, all validity and consistency checks are performed on the sample definition. If any errors are detected, message windows will be displayed indicating the nature of the error and suggested corrective actions. Once the sample data is judged error-free, then the save dialog window will be displayed for the user to specify the name of the sample file. All the tables associated with the table will be saved. An informative message window is displayed showing the sample tables being created.
- **Delete Sample** — deletes the currently selected sample. A confirmation message window is displayed to request user confirmation before the delete is performed.

The **Data Type** pull-down menu contains commands that allow the user to specify the type of information to be currently displayed on the screen for input or editing. It contains the following commands.

- **Description** — generates the description dialogue window.
- **Means and Std Dev** — generates the means and standard deviation dialogue window.

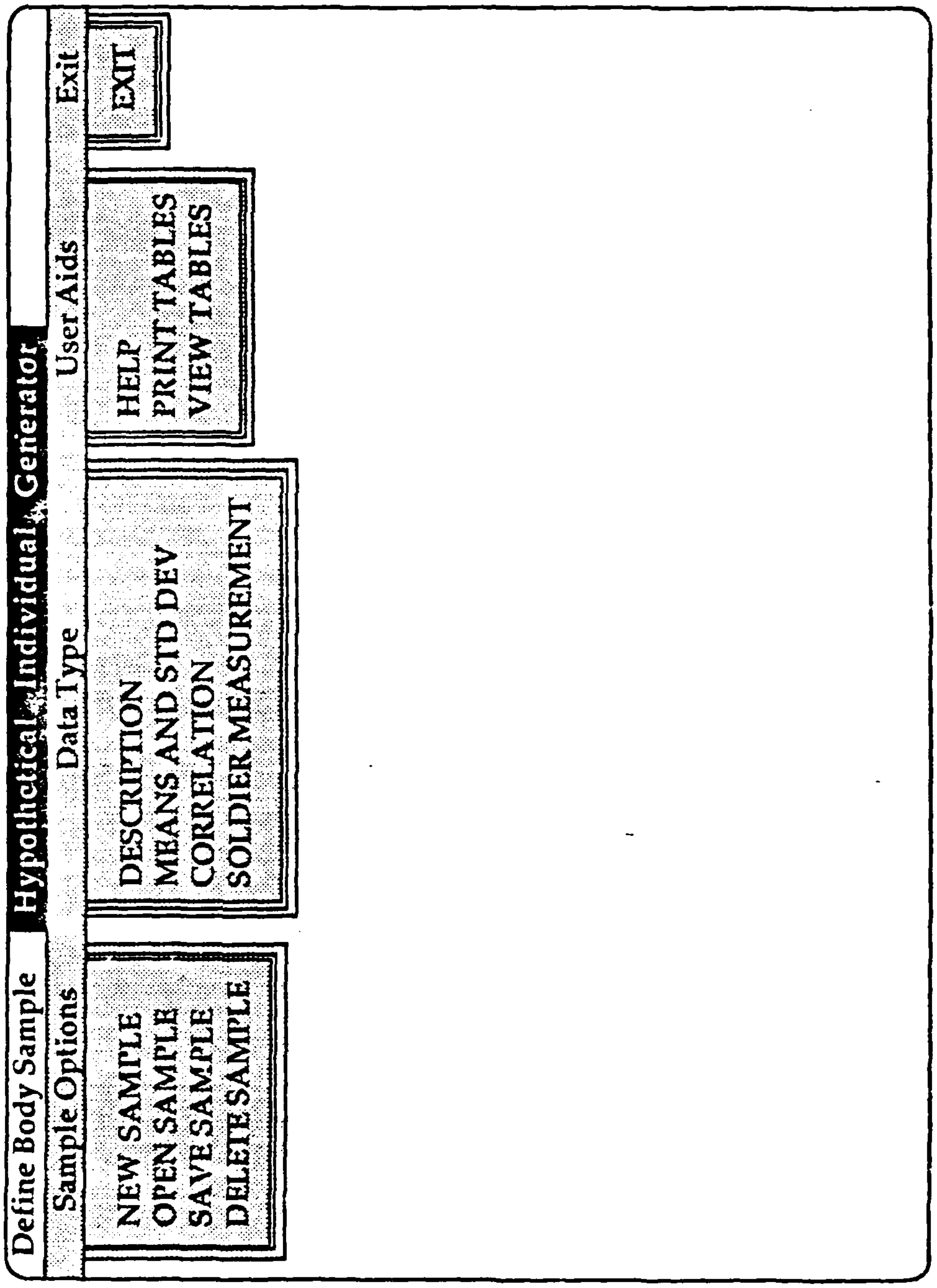


Figure 5-4: Body/Performance Sample Main Screen

- **Correlation** — generates the correlation dialogue window.
- **Soldier Measurement** — generates the soldier measurement dialogue window.

The **User Aids** pull-down menu contains the following commands:

- **Help** command that allow the user to obtain help windows about the BODYSAMP module.
- **Print Tables** command that allows the user to obtain a copy of any of the body tables on the printer.
- **View Data** command that allows the user to obtain a copy of any of the body sample tables on the screen.

BODYSAMP Windows.

The BODYSAMP sample description window is used for entering the necessary information to describe a sample. The current activity area of the title bar will contain the words 'Define Body Sample'. The sample description window, as illustrated in Figure 5-5, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the sample name field.
- **Save** — saves the current sample description as displayed on the screen. The following validation is performed prior to the actual saving the sample description:

  The sample name and description fields are valid names, i.e., they start with an alphabetic character (a-z) and do not contain any illegal characters.

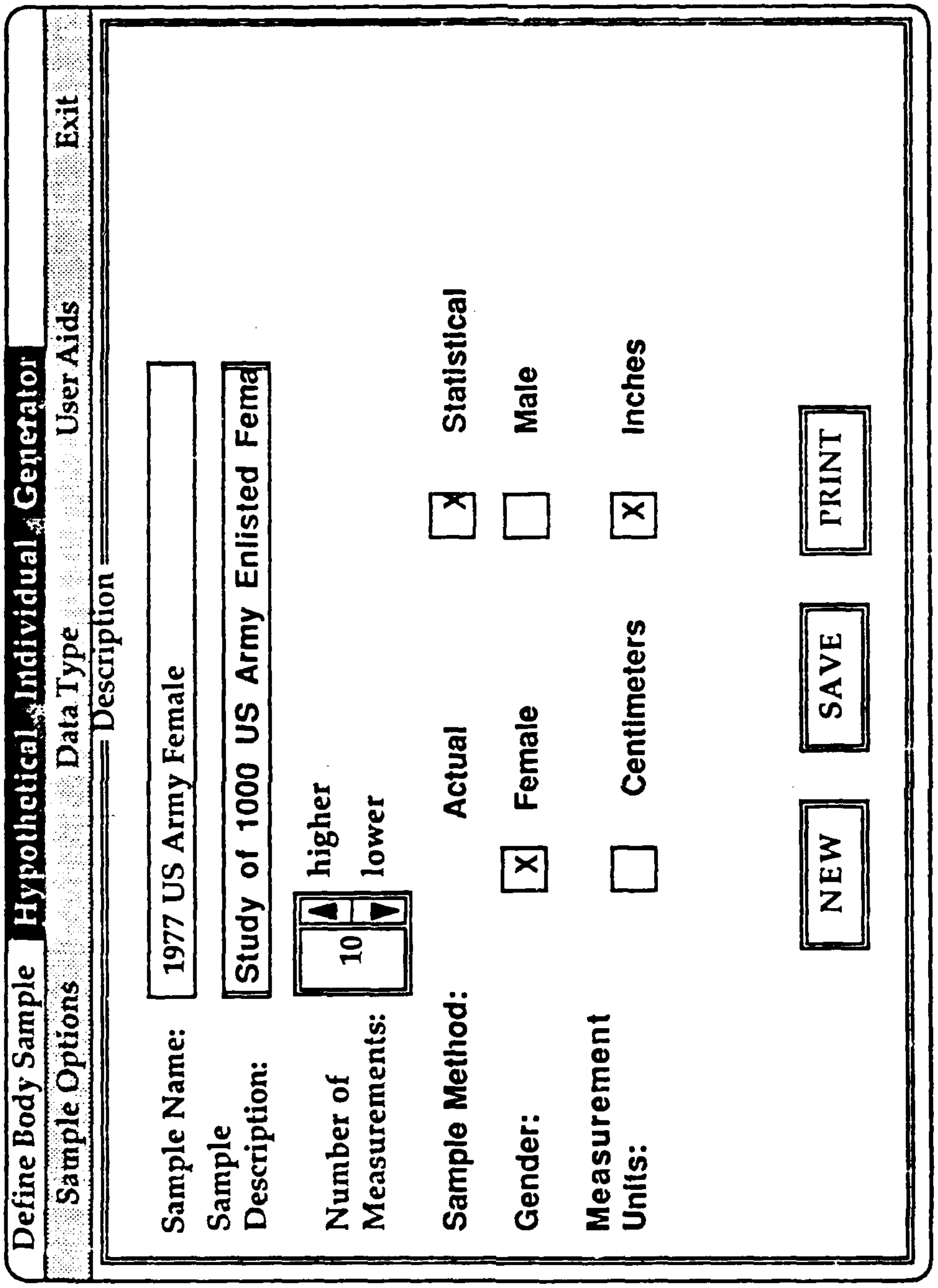
  The number of measurements is an integer number between 1 and 99.
- **Print** — print the current sample description on the printer.

The sample dialogue window contains the following text entry boxes:

- **Sample Name** — entry of the sample name as a maximum of 28 characters.
- **Sample Description** — entry of the sample description as a maximum of 80 characters.
- **Number of Measurements** — entry of the number of measurements included in the sample as an integer number between 1 and 99.

The following check boxes are used in the sample dialogue window:

- **Sample Method** — specify how the sample is to be generated as either 'Actual' or 'Statistical'.
- **Gender** — specify the sample gender as either 'Male' or 'Female'.
- **Measurement Units** — specify the units of measurements as either 'inches' or 'centimeters'.



Sample Name: 1977 US Army Female

Sample Description: Study of 1000 US Army Enlisted Fema

Number of Measurements: 10 ▲ higher ▼ lower

Sample Method: ☐ Actual ☒ Statistical

Gender: ☒ Female ☐ Male

Measurement Units: ☐ Centimeters ☒ Inches

Figure 5-5: Sample Description Windows.

The BODYSAMP means and standard deviation window is used for entering the mean and standard deviation information for a sample. The current activity area of the title bar will contain the words 'Define Body Sample.' The status information area contains the word "inches" to inform the user of the selected measurement units to be used for all input data. The means and standard deviation window, as illustrated in Figure 5-6, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the measurement identifier field.
- **Save** — saves the current means and standard deviation data as displayed on the screen. The following validation is performed prior to the actual saving the sample description:

    The sample name is a valid name, i.e., it starts with an alphabetic character (a-z) and does not contain any illegal characters.

    The measurement identifier is valid and contained in the standard measurement description table.

    The means of each measurement is in the range of the minimum and maximum values.

    The standard deviations cannot be negative.

    The mean must be greater than the standard deviation for each measurement.

- **Print** — print the current set of means and standard deviation data on the printer.

The mean and standard deviation dialogue window contains a set of text entry boxes arranged in rows and columns with the columns labeled for the fields described below and a row for each of the indicated number of measurements:

- **Measurement Identifier** — entry of an integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- **Measurement Description** — A 40-character text description that describes the measurement that is automatically filled in from the Standard Measurements table based upon the entered measurement identifier.
- **Mean** — entry of a real number that contains the means for the measurement.
- **Standard Deviation** — entry of a real number that contains the standard deviation for the measurement.
- **Minimum Value** — entry of a real number that contains the lowest acceptable value for the measurement. If not entered by the user, BODYSAMP will use the value of the mean minus three standard deviations.
- **Maximum Value** — entry of a real number that contains the highest acceptable value for the measurement. If not entered by the user, BODYSAMP will use the value of the mean plus three standard deviations.

The BODYSAMP correlation window is used for entering the intercorrelation matrix for a sample. The current activity area of the title bar will contain the words 'Define Body Sample'. The status information area contains the word 'Inches' to inform the user of the selected measurement units to be used for all input data. Since the correlation matrix is symmetric about the diagonal, only the upper triangular portion need be entered so that each successive line of data contains one less entry. For example, the user has specified fourteen measurements. The first row of the correlation matrix will require fourteen entries; the second row will require thirteen entries; row fourteen wiii require one entry. It is also assumed that the first entry in each row is one (1.0). The correlation window, as illustrated in Figure 5-7, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the measurement identifier field.
- **Save** — saves the current correlation matrix as displayed on the screen. The following validation is performed prior to the actual saving the sample description:

  The sample name is a valid names, i.e., it starts with an alphabetic character (a-z) and does not contain any illegal characters.

  The correlation values are between 0 and 1.
- **Print** — print the current correlation matrix on the printer.

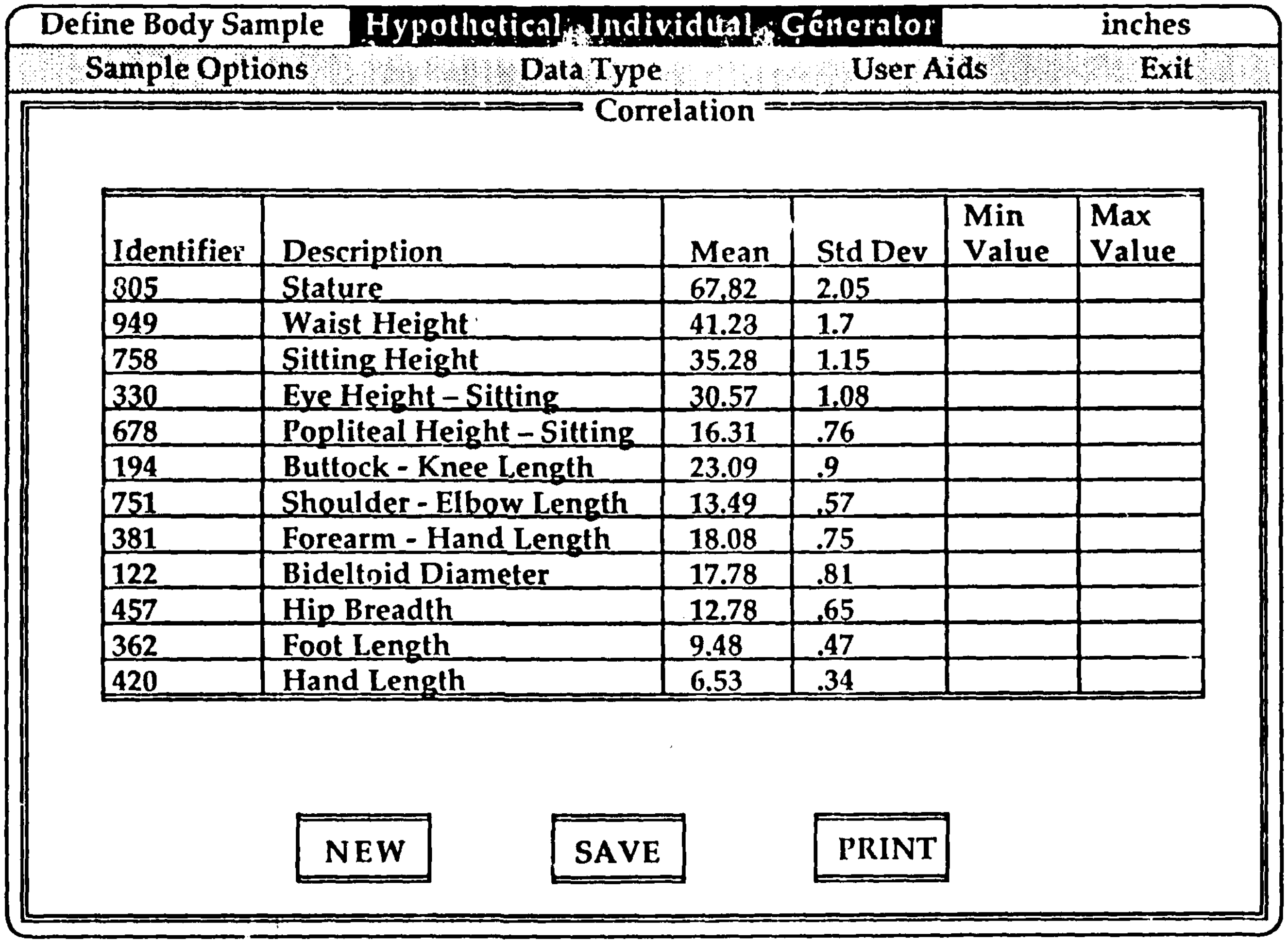
The correlation dialogue window contains a set of text entry boxes arranged in rows and columns. The entires in the Identifier column will be initially blank and the user will be required to enter the measurement identifier for the measurement correlation to be contained in row. As the user enters the identifier in row of the Identifier column, the column heading for the correlations will also be changed. A row is required for each of the indicated number of measurements:

- **Measurement Identifier** — entry of an integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- **Correlation Set** — entry of a set of real number that contains the means for the measurement. The first value in each row will be predefined as 1.0. The number of columns for each row will be equal to the number of measurements minus the row number.

The BODYSAMP soldier measurement window is used for entering actual soldier measurements for a sample. The soldier measurement window, as illustrated in Figure 5-8, contains the following pushbuttons:

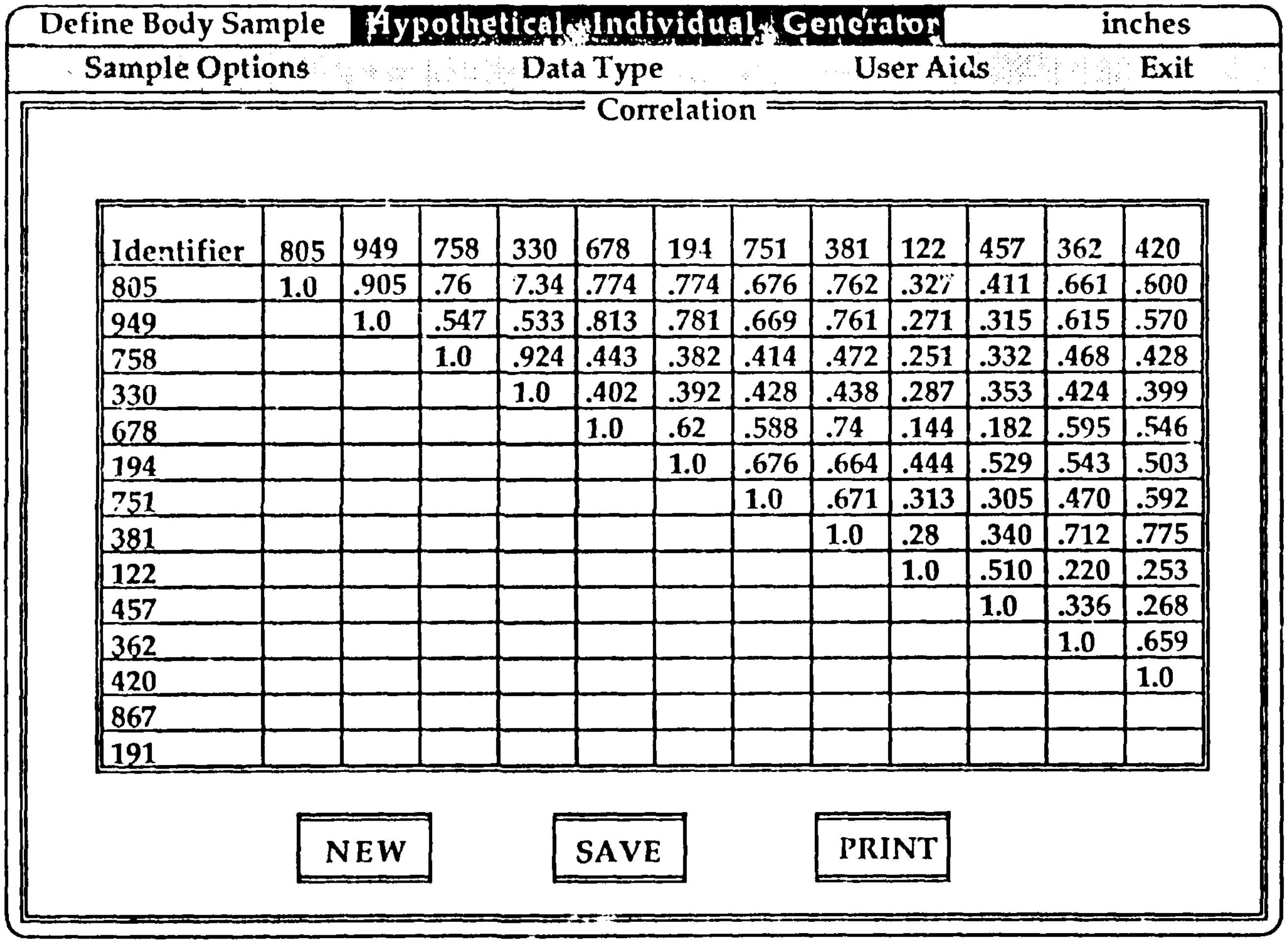
- **New** — clears all input fields and places the text cursor in the soldier id field.
- **Save** — saves the measurement data as displayed on the screen for the current soldier. The following validation is performed prior to the actual saving the sample description:

  The sample name is a valid names, i.e., it starts with an alphabetic character (a-z) and does not contain any illegal characters.



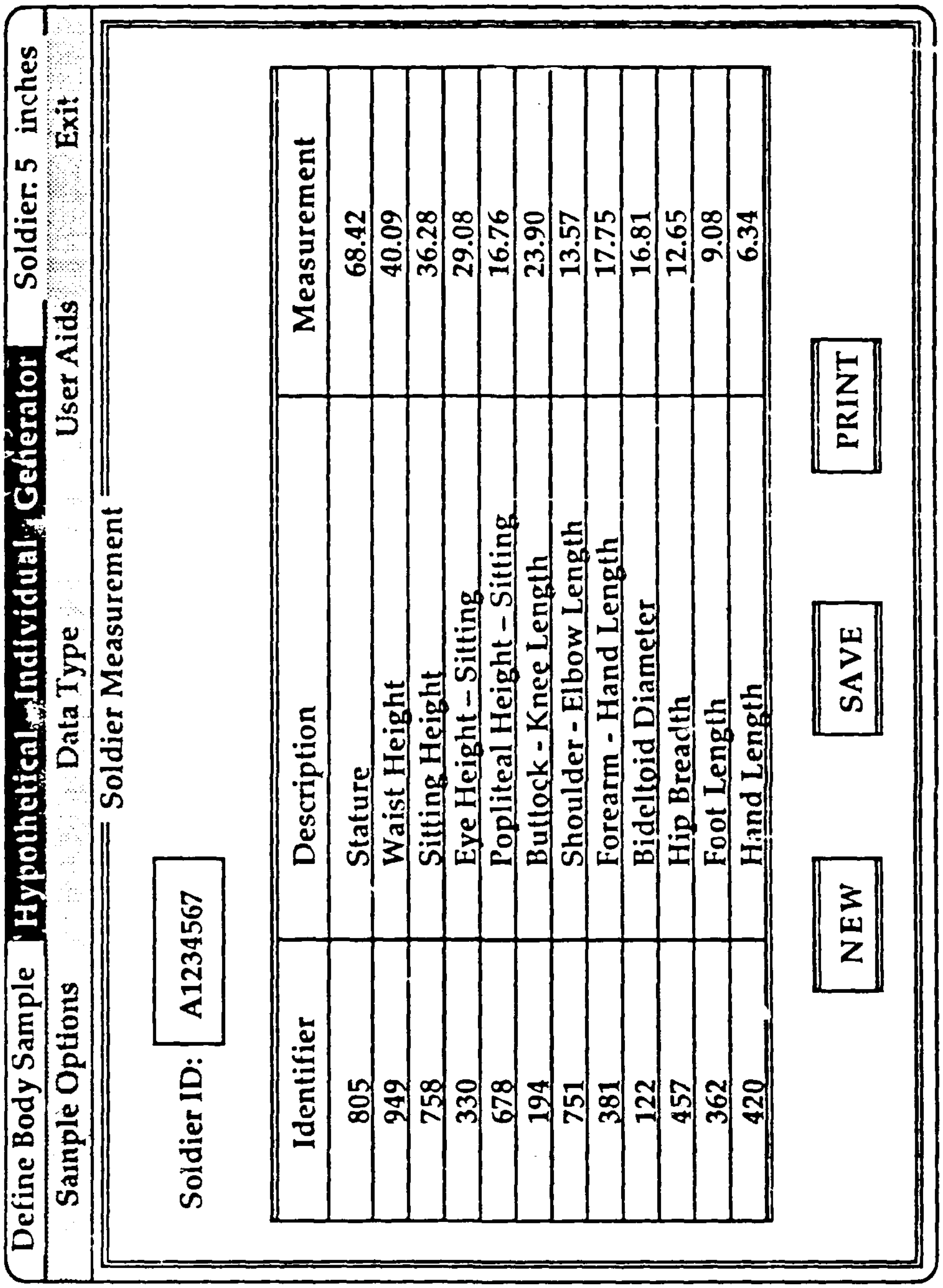
Identifier	Description	Mean	Std Dev	Min Value	Max Value
805	Stature	67.82	2.05		
949	Waist Height	41.23	1.7		
758	Sitting Height	35.28	1.15		
330	Eye Height – Sitting	30.57	1.08		
678	Popliteal Height – Sitting	16.31	.76		
194	Buttock - Knee Length	23.09	.9		
751	Shoulder - Elbow Length	13.49	.57		
381	Forearm - Hand Length	18.08	.75		
122	Bideltoid Diameter	17.78	.81		
457	Hip Breadth	12.78	.65		
362	Foot Length	9.48	.47		
420	Hand Length	6.53	.34		

Figure 5-6: Means and Standard Deviation Window.



Identifier	805	949	758	330	678	194	751	381	122	457	362	420
805	1.0	.905	.76	7.34	.774	.774	.676	.762	.327	.411	.661	.600
949		1.0	.547	.533	.813	.781	.669	.761	.271	.315	.615	.570
758			1.0	.924	.443	.382	.414	.472	.251	.332	.468	.428
330				1.0	.402	.392	.428	.438	.287	.353	.424	.399
678					1.0	.62	.588	.74	.144	.182	.595	.546
194						1.0	.676	.664	.444	.529	.543	.503
751							1.0	.671	.313	.305	.470	.592
381								1.0	.28	.340	.712	.775
122									1.0	.510	.220	.253
457										1.0	.336	.268
362											1.0	.659
420												1.0
867												
191												

Figure 5-7: Correlation Window.



Soldier ID: A1234567

Identifier	Description	Measurement
805	Stature	68.42
949	Waist Height	40.09
758	Sitting Height	36.28
330	Eye Height - Sitting	29.08
678	Popliteal Height - Sitting	16.76
194	Buttock - Knee Length	23.90
751	Shoulder - Elbow Length	13.57
381	Forearm - Hand Length	17.75
122	Bideltoid Diameter	16.81
457	Hip Breadth	12.65
362	Foot Length	9.08
420	Hand Length	6.34

Figure 5-8: Soldier Measurement Window.

The measurement identifier is valid and contained in the standard measurement description table.

The value of each measurement is in the range of the minimum and maximum values obtained from a reference population.

When the save operation is completed, all fields in the dialogue window will be cleared and the text cursor placed in the soldier id field.

- **Print** — print the current set of soldier measurement data on the printer.

The status information area of the title bar will contain the sequential integer number of the soldier in the sample currently being processed starting from one. The actual soldier measurement window contains a text entry box for the user to specify the **Soldier ID**. Beneath the **Soldier ID** box will be a set of text entry boxes arranged in rows and columns with the columns labeled for the fields described below and a row for each of the indicated number of measurements:

- **Measurement Identifier** — entry of an integer number that identifies the measurement obtained from the NASA Source Book (NASA, 1978).
- **Measurement Description** — An 40 character text description that describes the measurement that is automatically filled in from the Standard Measurements table based upon the entered measurement identifier.
- **Measurement** — entry of a real number that contains the value for the measurement.

5.2.4 PERFSAMP

5.2.4.1 Description

The PERFSAMP module of HIG maintains information for all user-defined samples of performance measurements. When defining soldier characteristics, the performance measures of individual soldiers can be input directly into HIG or HIG can generate a sample of soldiers from statistics. If the user desires the latter option, HIG requires input of the means, standard deviations, and an intercorrelation matrix of the set of performance measures. In order to clearly identify the performance measurements, a numeric code will be used to identify measurements The current set of performance measurements are defined in Appendix F.

PERFSAMP maintains the database tables containing the sample description, means and standard deviations, intercorrelation matrices, sample measurements, and standard measurements descriptions. The data base tables will be identical to that used for the Body sample as described in section 5.2.3.2 with the exceptions noted below. The Sample Type field in the Sample Description table (SAMPDESP) will containing the word 'Performance' to indicate the type of soldier data. A separate Standard Measurements Description table (PMEASDES) will contains the following elements for each standard performance measurement:

- Measurement Identifier: An integer number that uniquely identifies the measurement.
- Measurement Name: A 40 character text description that names the measurement.
- Measurement Description: A 200 character text description that describes the measurement.

The list of performance measurements are described in Appendix F.

5.2.4.2 PERFSAMP Screens

The PERFSAMP screens and user definition of the performance sample will be identical to those used for BODYSAMP as described in Section 5.2.3.3 except the words 'Define Performance Sample' will be inserted in the current activity area of the title bar.

5.2.5 GENSAMP

5.2.5.1 Description

After the user has defined the parameters for the desired soldier population, GENSAMP will generate a sample of hypothetical individuals with either the body and/or performance measurements. Using a Monte Carlo technique, GENSAMP then constructs a sample of soldiers whose measures reflect those of the population (100 soldiers are generally constructed). Since the sampling technique follows the rules of the intercorrelation matrix, the final set of soldiers will have correctly proportioned measurements. A set of normally distributed random numbers generated using the Marsaglia-Bray algorithm with the gate technique. This set of normally distributed random numbers with a mean of zero and a standard deviation of one are generated and applied to the intercorrelation matrix to produce measurements with intercorrelations that are in agreement with the input intercorrelation matrix. The algorithm is summarized below:

> Select means to create "average individual." $V(I) = m(I)$.where $I = 1$ to NVAR.
>
> Generate SV(I) vector by using the square root of R multiplied by a randomly generated vector of z values.
>
> Use means plus a randomly selected value, (in SV(I) vector) to create other hypothetical individuals. $V(I) = m(I) + SV(I) * s(I)$ where $I = 1$ to NVAR.

where (Note: capital letters are used to represent a matrix or vector; lower case to represent the entries in the matrix or vector):

R(I,J) = Representation of the correlation matrix .

M(I) = Means, m, for the original variables.

S(I) = Standard deviations, s, for original variables.

V(I) = Values for one HI generated in this routine. Represents scores from Project A data base for a hypothetical individual. (This vector is reused.)

SV(I) = Standard score representation of the means. Used in formula 2 to adjust means to create values for a Hypothetical Individual.

ISUB = Subscript representing the Hypothetical Individuals sampled, from 0 to NCASES where NCASES is defined as the number of Hypothetical Individuals to be generated. Default of NCASES is 100, meaning 101 cases are generated.

ISUB = 0 is the subscript for the average (centroid) case.

GENSAMP maintains the SAMPLE table containing the sample description and sample measurements for the set of hypothetical individuals. The elements of the database tables are described in Section 5.2.3.2.

5.2.5.2 GENSAMP Screens

The GENSAMP screen initially contains the title and menu bars. As the user selects various menu options, the window portion of the GENSAMP screen will contain the appropriate dialog windows for the user to enter the indicated information.

GENSAMP Title Bar. The title bar of GENSAMP contains the words 'Hypothetical Individual Generator' as the function name. The words 'Generate Sample' will be inserted in the current activity area of the title bar. The status information areas of the title bar will contain various information depending upon the current activity.

GENSAMP Menu Bar. GENSAMP contains the following menu options of the menu bar as illustrated in Figure 5-9:

- **Sample Type** — indicates type sample as either Body or Performance.
- **Generate Options** — specifies either generate a sample or view the generated sample data.
- **User Aids** — provides the capabilities to print sample information or obtain help messages.
- **Exit** — terminates GENSAMP and returns the user to the HIGMAIN main menu.

The **Sample Type** pull-down menu contains commands that allow the user to specify the type of information to be included in the generated sample. It contains the following commands:

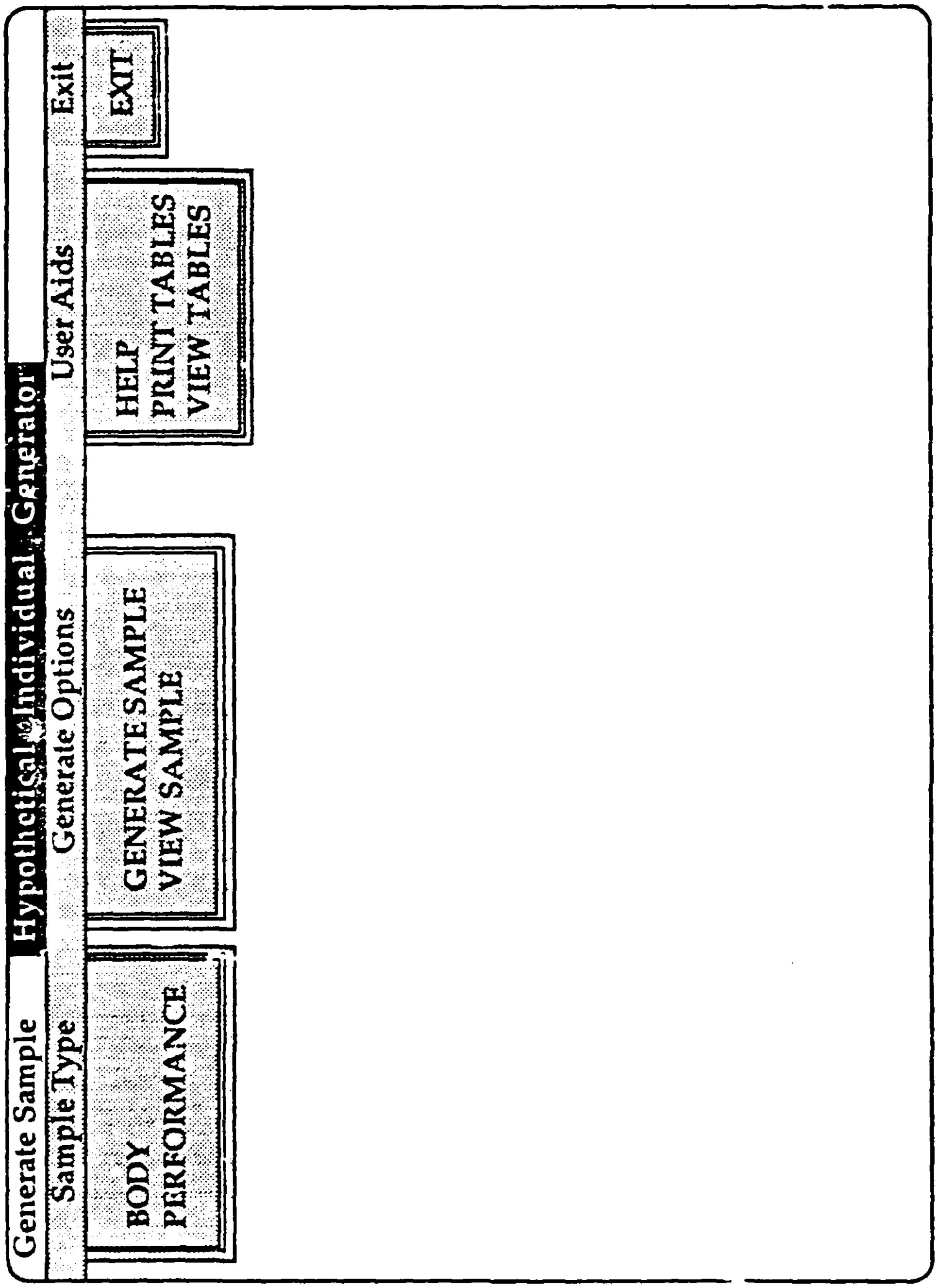


Figure 5-9: Generate Sample Main Screen.

- **Body** — the generated sample will contain body measurements.
- **Performance** — the generated sample will contain performa-ce measurements.

The **Generate Options** pull-down menu contains commands that allow the user to create and view the generated sample. It contains the following commands:

- **Generate Sample** — indicates that a new set of hypothetical individuals should be generated. If a sample is currently open, it will close the current sample after asking if the changes should be saved and then creates a new empty sample. It clears the current dialogue window and puts the generate dialogue window on the screen with the text cursor placed in the first character in the sample size field.
- **View Sample** — opens the tables associated with an existing sample. It closes the current sample after asking if changes should be saved and then opens the sample file selection dialog box. The user views the currently defined sample names and then selects one. The screen is cleared and the selected sample information is placed in a table viewing window that allows the user to scroll through the contents of the table. The text cursor is placed in the first character in the table

The **User Aids** pull-down menu contains the following commands:

- **Help** commands that allow the user to obtain help windows about the GENSAMP module.
- **Print Tables** command that allows the user to obtain a copy of any of the sample files on the printer.
- **View Tables** command that allows the user to obtain a copy of any of the sample files on the screen.

GENSAMP Windows.

The GENSAMP generate window is used for entering the necessary information to generate a sample. The current activity area of the title bar will contain the words 'Generate xxx Sample' with xxx indicating either Body or Performance. The generate window, as illustrated in Figure 5-10, contains the following pushbutton:

- **Generate** — saves the current generation information as displayed on the screen and then generates the set of hypothetical individuals. The following validation is performed prior to the actual saving the sample description:

  The sample is between 1 and 400.

  The random number seed is between 1 and 9.

The generate window contains the following text entry boxes:

- **Sample Name** — entry of the sample name as a maximum of 28 characters.

The generate window contains the following click boxes:

- **Sample Size** — specification of the size of the sample as between 1 and 400.
- **Random Number Seed** — specification of the random number seed.

A list selection box showing the list of available sample files for the selected sample type is displayed in the upper left comer of the generate dialog window as shown in Figure 5-10.

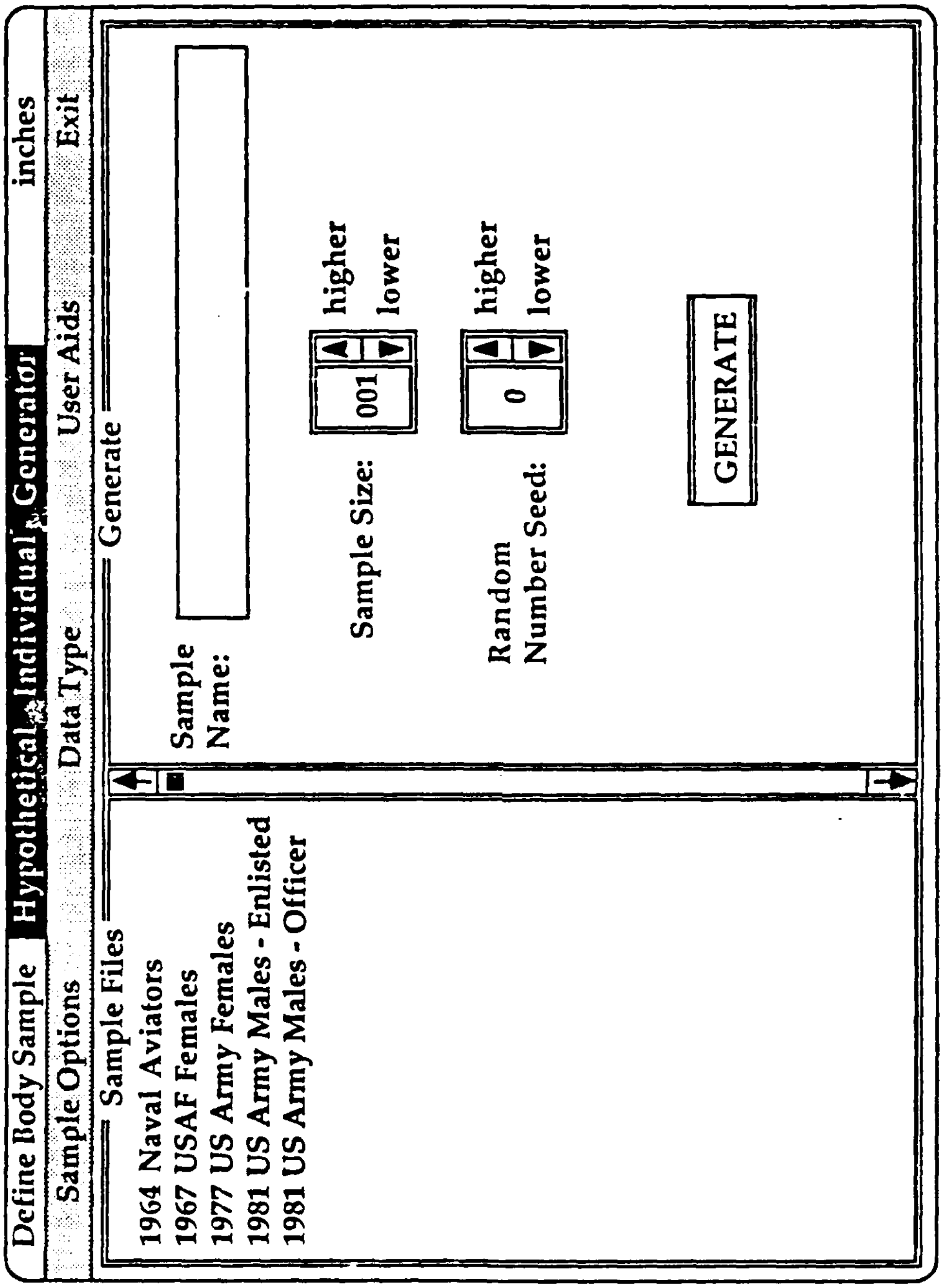
GENSAMP Message Window.

The GENSAMP message window shown in Figure 5-11 is displayed indicating which member of the sample is currently being generated and allowing the user to use the Cancel pushbutton to terminate the generation process.

5.3 Define Systems

Activities external to the soldier are treated as events. This means system representation will be established in the context of discrete events. While this might seem to be a handicap, many operator tasks and most maintainer tasks are discrete. Facilities for creating events are contained in the dynamic simulation.

Previous versions of HOS contained provisions for dynamic simulation of systems. At present, HOS-IV does not have a facility for communicating with a simulation such as might be desired for helicopters, etc. nor is it anticipated that such a facility would be included in SEPTA. Because on-line simulations have been accomplished before, inclusion of such a facility will not be difficult conceptually. Briefly, we conceptualize a simulation being done on another computer connected to the computer running the dynamic simulation, with parameters, responses, information, etc. being passed back and forth between the two computers.



Sample Files

1964 Naval Aviators
1967 USAF Females
1977 US Army Females
1981 US Army Males - Enlisted
1981 US Army Males - Officer

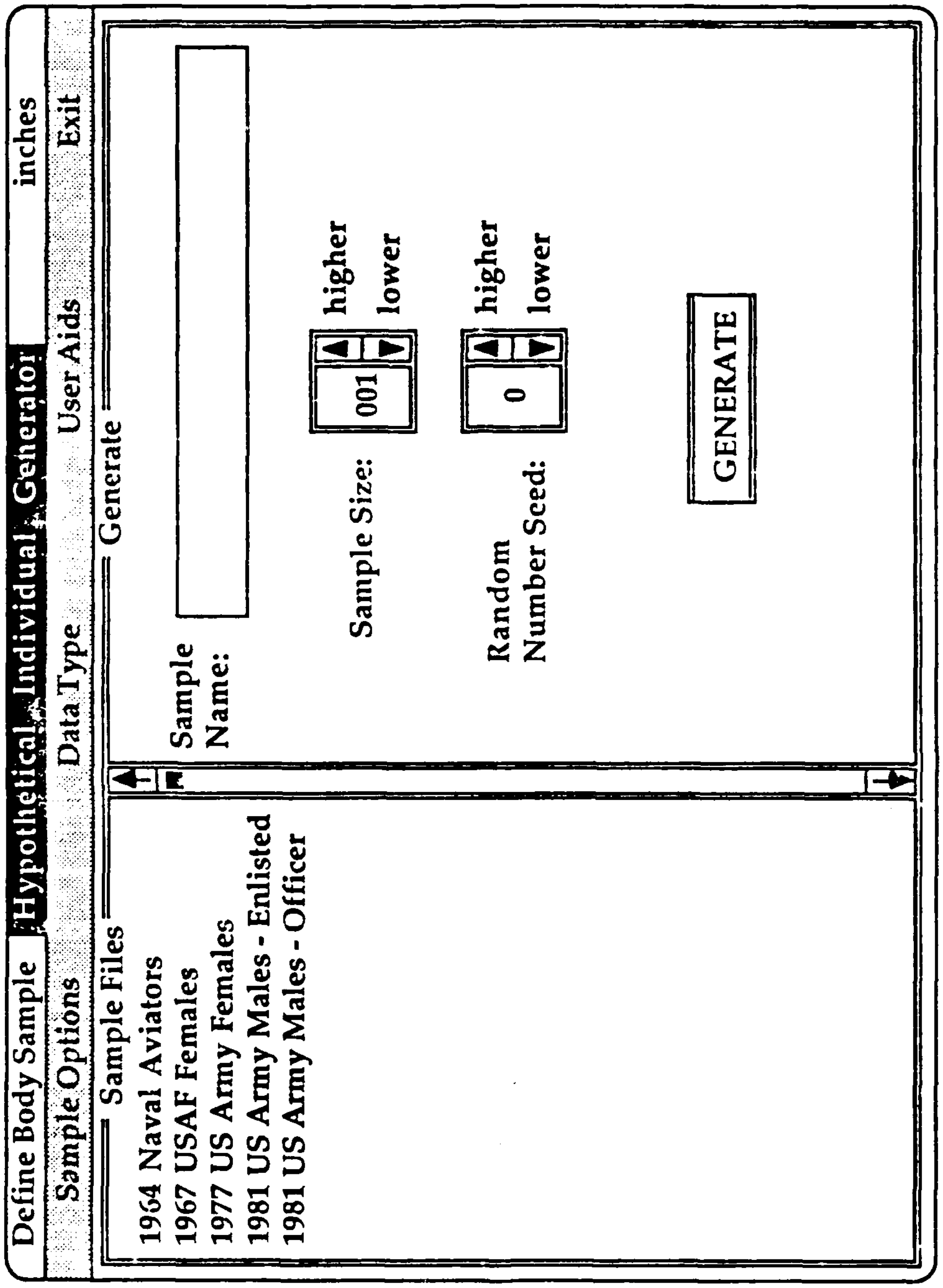
Generate

Sample Name:

Sample Size: 001 ▲ higher ▼ lower

Random Number Seed: 0 ▲ higher ▼ lower

Figure 5-10: Generate Window.



1964 Naval Aviators
1967 USAF Females
1977 US Army Females
1981 US Army Males - Enlisted
1981 US Army Males - Officer

Sample Name:

Sample Size: 001

Random Number Seed: 0

Figure 5-11: Generate Measures Window.

5.4 KUIP Entry – Key Underlying Internal Processes

The terms micromodels and underlying internal processes (UIPs) have been used interchangeably. A micromodel can be defined as a detailed model and description of a specific behavior. It corresponds most closely to the definition of Task Element presented in Table 3-1 showing a structured task analysis outline. We differentiate here between UIPs and Key UIPs. The differentiation is a simple one: UIPs are used in most behavior. Key UIPs are specifically derived from modeling and simulating performance on personnel tests used by the Army. The Key Underlying Internal Processes (KUIPs) provide the link between the personnel characteristic measurement data and system performance simulation. KUIPs are special UIP - micromodels derived solely from soldier characteristic data, e.g., Project A. These KUIPs serve a dual purpose: they may be used as any other micromodel in evaluating soldier performance while maintaining or operating the system and they are used to define performance in terms of personnel characteristics. Additionally, values can be modified or added to the KUIP tables which will be described below along with supplementary capabilities and modification capabilities which will be developed in subroutines. The general flow of the process is shown in Figure 5-12.

Micromodel Definition

Micromodel: A detailed and generalized description of the specific behavior to be modeled; a model of one or more Underlying Internal Processes. For example, SHORT-TERM MEMORY would be a micromodel as would GRASPING. A software procedure makes a call to a Micromodel passing parameters about the object to be grasped. The micromodel employs these parameters to simulate performance.

In actual operation, KUIP generation does not differ from creating any other type of dynamic simulation. Indeed, most of the software components required are contained in the dynamic simulation and some of that material is presented here for convenience. The inclusion of separate modules for KUIP Generation is to insure the conceptual distinction for purposes of SEPTA between performance descriptions for personnel characteristics and performance descriptions for system maintenance and operation. Screens are shown in Figures 5-14 to 5-19 which will spawn the appropriate dynamic simulation modules.

There are some important differences between standard UIP-micromodels and KUIPs. Most of these differences will be transparent to the user and we would not anticipate the user creating any KUIP models. By the conclusion of the MANPRINT Methods Product development we will have developed micromodels for all tests and data currently in the Project A database. The

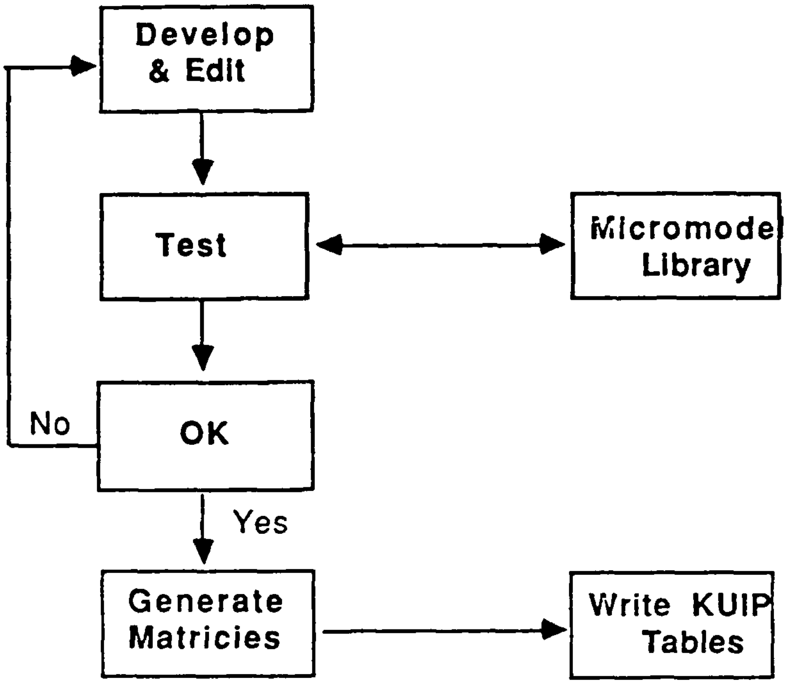


Figure 5-12: Representation of Entry of KUIP Micromodels and Values

development of these KUIPs will not be a trivial task. In order to be truly representative of human performance, the KUIP must not only produce the means and standard deviations shown in Table 1 of Appendix F but also produce the factor loadings shown in Table 2 of Appendix F. The analysis included in Appendix F shows the need for developing means to handle strategies as well as straight performance. However, the capability needs to be available for future developments in personnel testing

5.4.1 Knowledge Representation

Knowledge representation defines all the information pertinent to the soldier, system, and environment. The HOS-IV simulation requires definition of the following types of information:

- Objects
- Action Rules
- Actions
- Discrete Events

A description of each of the types of information follows.

5.4.1.1 Objects

The dynamic simulation uses an object-attribute structure to store data on each entity, or object, to be modeled in the simulation (e.g., displays, controls, sensors, aircraft). Each object has an associated list of attributes (e.g., size, location) and each attribute is assigned a value. These attributes describe the important features or characteristics of an object. Since the term 'attribute' was considered difficult for users to understand, the term characteristics is used instead. Values indicate the state of the characteristic at a particular point in the simulation. Objects can be defined at various levels of detail. For example, the object EMERGENCY INDICATOR LIGHT can have one characteristic, STATUS, with a value of ON or OFF. Another object may have several characteristics, such as ENEMY EMITTER with characteristics for LOCATION, SPEED, BEARING, FREQUENCY, PULSE WIDTH, PULSE REPETITION FREQUENCY, and PULSE MODE, along with their assigned values. The values of the characteristics of OBJECTs indirectly control the flow of events in the simulation by providing information to the RULES and ACTIONS.

An example of an object is as follows:

Object Name:	Radar
Characteristics:	
Status	on/off
Mode	automatic/manual
X-location	46.78
Y-location	124.54

In the example above, the user has defined the x and y crewstation coordinates of the radar control panel. The z coordinate is not important for this particular simulation so the user chose not to define it. Here, the user is interested in determining motor activity times, that is, the time required by the soldier micromodels to move their eyes and hands to the panel. Alternatively, the user may not define location and use fixed time charges for soldier motor activities.

The object-characteristic structure provides a powerful and efficient knowledge representation capability that provides the user with the flexibility to define what is important to the application at any level of detail.

5.4.1.2 Action Rules

Action rules drive what actions will be executed at a particular simulation time snapshot. These rules are defined by a starting and ending conditional, the name of the action to be invoked if the starting conditional is true, and a priority assignment from 0 (lowest) to 9 (highest). Separate queues are provided for the soldier, hardware, and environment. If the starting conditional statement is true, the named action will be invoked. The action will then continue until the ending conditional is true. Rules can be declared active or inactive during the simulation. At each simulation snapshot, the conditional statements of all active rules will be evaluated to determine what actions will be executed or terminated. The rules will be processed in the sequence specified by the priority assignment. This precedence relationship is particularly useful as actions invoked by a high priority rule may suspend lower priority rules. This capability permits rules to be categorized by mission critical tasks as well as mission phase. All active rules represent the processes that are occurring in parallel. If sequential processing mode is desired, the rules must be set up in such a fashion that only one rule is active at any particular time.

Each rule is defined by three components:

- An IF clause,
- A DO clause, and
- An UNTIL clause.

An example of a rule is:

IF status OF radar EQUALS on
DO process_emitters
UNTIL status OF simulation EQUALS off

This rule will be invoked only when the UNTIL clause is false and the IF clause is true.

5.4.1.3 Actions

Actions describe what will be done by the soldier, system, and environment at a given simulation snapshot if the related rule is true. The steps to accomplish a task which must be performed at a given mission time based on the current environmental and system status are detailed in the action. Actions can include updates to the values of object characteristics, invocation of other actions, and the initiation or suspension of action rules. Actions are the only simulation mechanism which can affect the values of the characteristics of objects.

Actions are defined using a small set of standard verbs (e.g., PERFORM, SET, SUSPEND) known as the HOS-IV Action Language (HAL). One of the major decisions in developing HAL involved whether to incorporate a true natural language capability into HOS-IV or to utilize a limited set of verbs. An unrestricted natural language capability would provide greater freedom of expression, but for HOS-IV simulations it does not appear to be necessary and in many cases would lead to greater ambiguity. Although unrestricted natural language is the least limited to the user, it is not currently practical to machine-parse on an IBM PC/AT level computer because of its extreme complexity. Additionally, a body of evidence suggests that limited the vocabulary and syntax available to the user improves the user's ability to utilize and comprehend the language (Bailey, 1985; Hendler & Michaelis, 1983). Therefore, it was decided to develop a limited set of verbs with restricted syntax to specify procedural knowledge. For HOS-IV, a small set of verbs which convey the required actions has been developed. Verbs are the 'primitive' building blocks with which the user constructs complex and application-specific actions.

One of the convenient features of HOS-IV is that models written in the C language can be directly integrated into the HOS-IV ACTION language. In order to embed a model of a radar processor which was written in C, the Key Word START_C_CODE is placed before the module and END_C_CODE at the end of the module. HAL provides full functionality in data manipulation and simulation processing.

5.4.1.4 Discrete Events

HOS-IV permits the definition of discrete events in order to simulate processes that are not originated by the soldier but which have an impact on the course of the simulation. Discrete events are defined by the event-name, the time of the event, and the action name to be executed. Typical events can include special tasking messages communicated to the soldier from a higher command, hardware system failures, or environmental changes (e.g., weather). The user defines the name of the event, time of event, and the action to be triggered. For example:

At Time:	00:05:00.000
Do:	system_failure
Event Name:	Electrical storm causes power outage.

5.4.1.5 System Interactions

Figure 5-13 illustrates the interactions and functional flow of information within HOS.

As illustrated in Figure 5-13, OBJECTs store values for all the entities to be simulated. The user partially controls the level of simulation detail by the number of objects and the type of characteristics defined for each object. For example, enemy aircraft can be a single object with an associated probability of a particular number coming within range of a radar at each simulation snapshot. Alternatively, 50 separate OBJECTS can be defined, each representing a unique

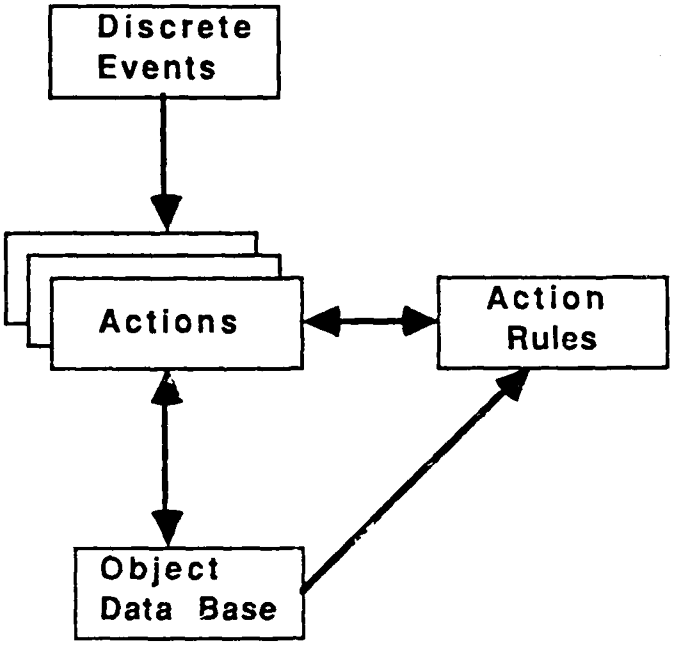


Figure 5-13: HOS-IV Simulation Structure and Functional Flow

aircraft with a specific geographic location relative to the radar system. The values of objects are dynamically controlled by ACTIONS which occur during the simulation. ACTIONS are triggered by a set of RULES. RULES define the conditions which will control the flow of events in the simulation.

Depending on the state of the object data base at each simulation time snapshot, RULES will trigger ACTIONS. ACTIONS in turn can suspend or activate RULES as well as modify the data base. Typically, RULES are grouped by mission phase. If, in an action, it is determined that a new mission phase has started, the ACTION will suspend a whole set of RULES and activate other RULES. DISCRETE EVENTS can also trigger ACTIONS.

5.4.2 HOS-IV Outputs

The HOS-IV outputs include:

- A timeline of events for the soldier, system, and environment,
- User-defined measures of effectiveness, and
- Standard analysis, such as:
  - Mean time to complete an action,

- Number of times an action is performed,
- Proportion of the soldier's time spent on each action.

HOS-IV allows for user-controlled detail of simulation outputs. When building the object library, the user can identify objects whose values are to be tracked throughout the simulation. User-defined measures of effectiveness, such as the number of contacts processed per hour or a history of error rates, can also be defined and stored in a separate simulation file at simulation end . With these data, the user can identify soldier and system bottlenecks and determine periods of soldier overload and the circumstances surrounding those overload periods. The models are described in detail in the HOS-IV User's Guide (Harris et al., 1988).

5.4.3 MicroModel Library

A library file will allow access to the micromodel database to facilitate finding, editing, or creating micromodels. Directly connected to this library or inventory will be a module which we will call a Bill of Behaviors (BOB), which will facilitate the user in constructing new micromodels. The way this will work will be to allow short-hand entries when constructing micromodels. The user will enter a micromodel name off the library/inventory list and the BOB module will fetch the micromodel. The user will then be able to use it as is or to modify it as appropriate. If modified, the user will be able to save the module in the master micromodel file for later use in another analysis. This BOB module cannot be designed until we have greater experience with the new dynamic simulation and have developed the overall structure of the human model.

5.4.4 Output

Text goes into the micromodel library file with special tags.

The principal output will consist of micromodels for various tests and measurements contained in the Project A data set. These codes go into a read-only file for protection. (The user will be able to copy the micromodels to a new file for modification.)

The frequency of use of each micromodel is entered into a table, KUIPFREQ, where columns represent different KUIPs and rows represent different tests.

KUIP^{-1} = transform matrix representing the of the KUIPFREQ table. This is used to adjust the KUIPs parameters into to be entered into values representing the KUIPs for an individual hypothetical individual.

The entries are frequencies for i = 1 to NKUIP, where NKUIP represents the total number of KUIPs, K = 1 to NTEST where NTEST is the total number of tests simulated, and kf = frequency of usage of each KUIP which is used as a transformation constant between the KUIP parameters and the personnel characteristic scores from Project A. Table 5-4, KUIPFREQ, is used in the analysis program to evaluate performance of each hypothetical individual. The details of the mathematical manipulations will be discussed in Section 8.

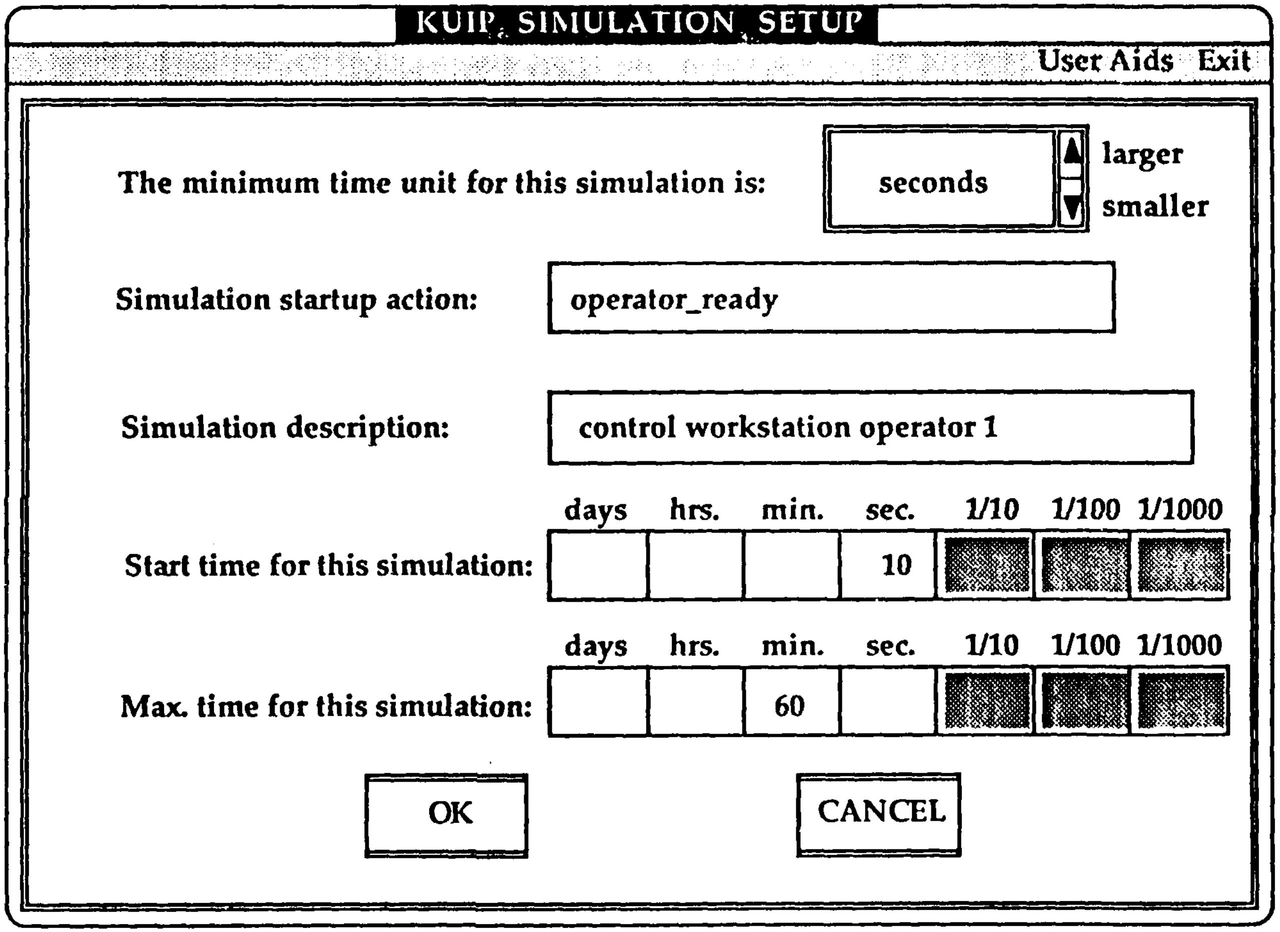
Table 5-4: The KUIP Frequency Transform Matrix Representation for KUIP Time and Accuracy Adjustments.

	1	2	3	4	...	...	...	I	Mean
1	kf_{11}	kf_{12}	kf_{13}	kf_{14}				kf_{1i}	X_1
2	kf_{21}	kf_{22}	kf_{23}	kf_{24}				kf_{2i}	X_2
3	kf_{31}	kf_{32}	kf_{33}	kf_{34}				kf_{3i}	X_3
.									
.									
.									
K	kf_{k1}	kf_{k2}	kf_{k3}	kf_{k4}				kf_{ki}	X_k

The entries in Table 5-4 are parameters for i = 1 to NKUIP, where NKUIP represents the total number of KUIPs, K = 1 to NTEST where NTEST represents the total number of tests simulated, and ku represents the parameter for each KUIP. These parameters can be performance times or accuracies; the two will have to be treated separately according to the general outlines of UIP theory. These two tables, KUIPTIME and KUIPACC, (Table 5-5) are used in the dynamic simulation as parameters for the KUIP micromodels.

Table 5-5: The KUIP Parameter Matrix Representation for KUIP Application. (Separate tables are used when ku = time and when ku = accuracy.) .

	1	2	3	4	...	...	...	I
1	ku_{11}	ku_{12}	ku_{13}	ku_{14}	...	...		ku_{1i}
2	ku_{21}	ku_{22}	ku_{23}	ku_{24}	...	...		ku_{2i}
3	ku_{31}	ku_{32}	ku_{33}	ku_{34}	...	...		ku_{3i}
.	.							
.	.							
.	.							
K	ku_{k1}	ku_{k2}	ku_{k3}	ku_{k4}	...	...		ku_{ki}



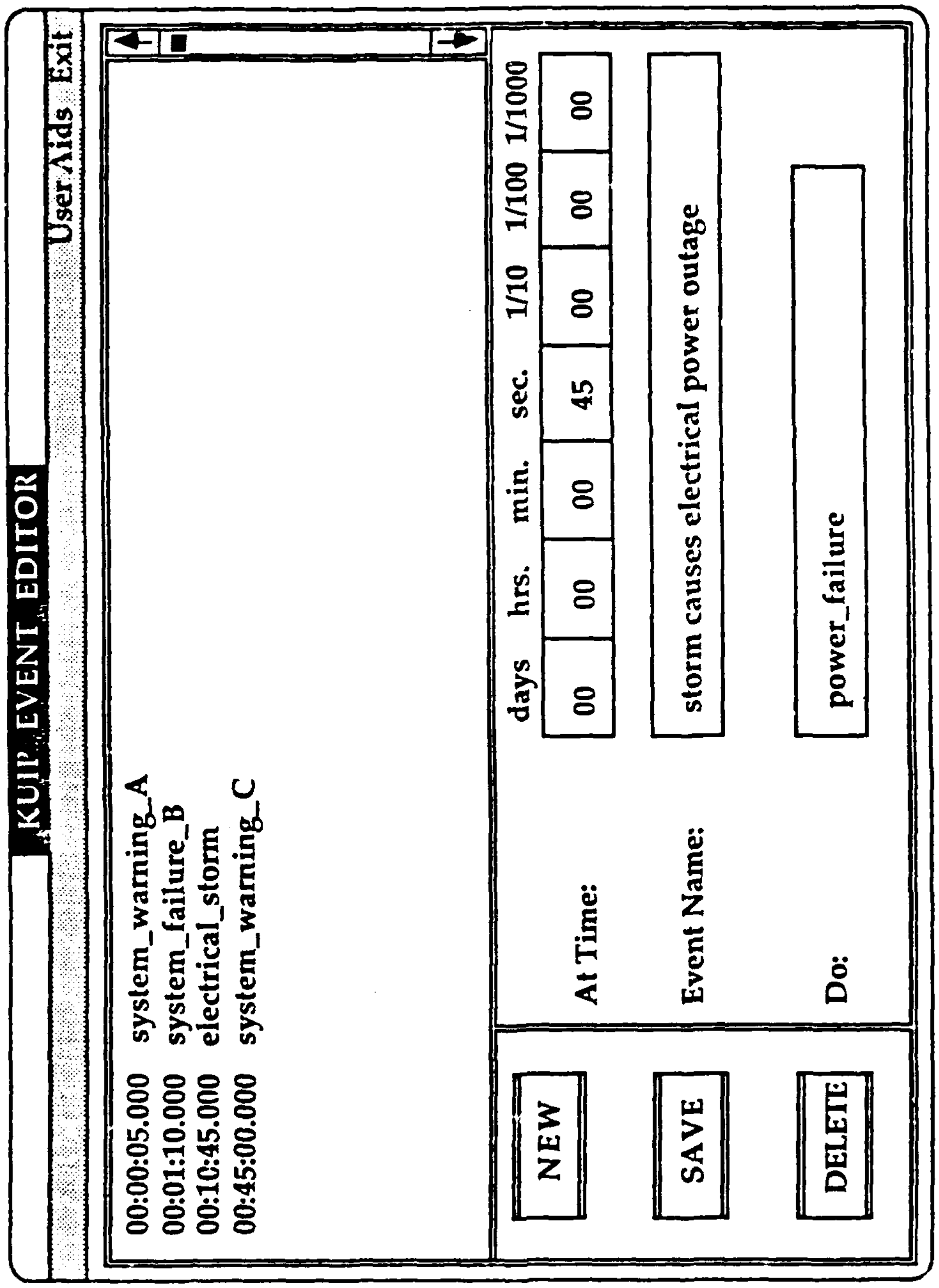
The minimum time unit for this simulation is: seconds ▲ larger ▼ smaller

Simulation startup action: operator_ready

Simulation description: control workstation operator 1

	days	hrs.	min.	sec.	1/10	1/100	1/1000
Start time for this simulation:				10	▨	▨	▨
Max. time for this simulation:			60		▨	▨	▨

Figure 5-14: KUIP Simulation Setup Screen.



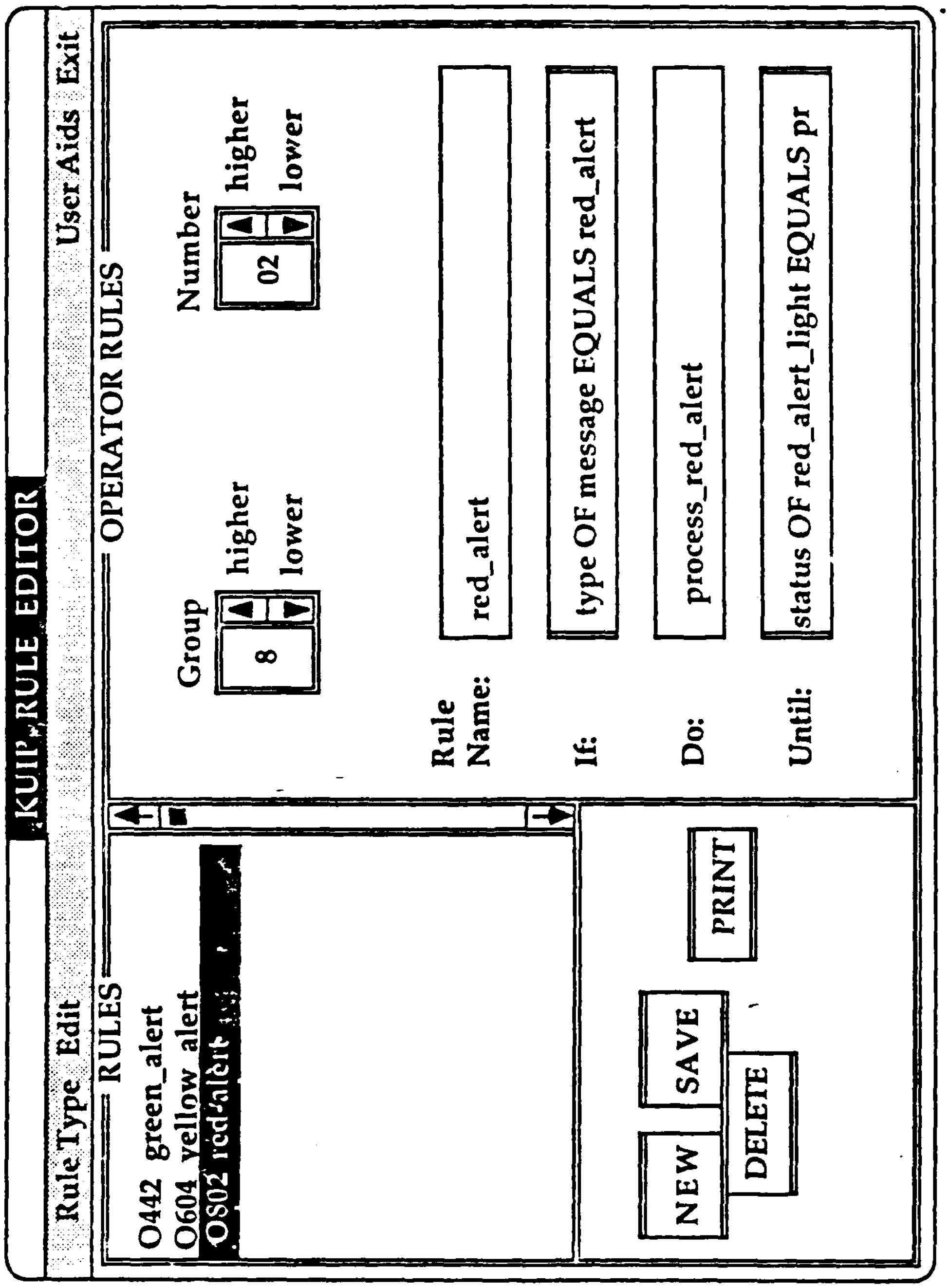
```
00:00:05.000  system_warning_A
00:01:10.000  system_failure_B
00:10:45.000  electrical_storm
00:45:00.000  system_warning_C
```

	days	hrs.	min.	sec.	1/10	1/100	1/1000
At Time:	00	00	00	45	00	00	00

Event Name: storm causes electrical power outage

Do: power_failure

Figure 5-15: KUIP Event Editor Screen.



O442 green_alert
O604 yellow_alert
O802 red_alert

Group: 8

Number: 02

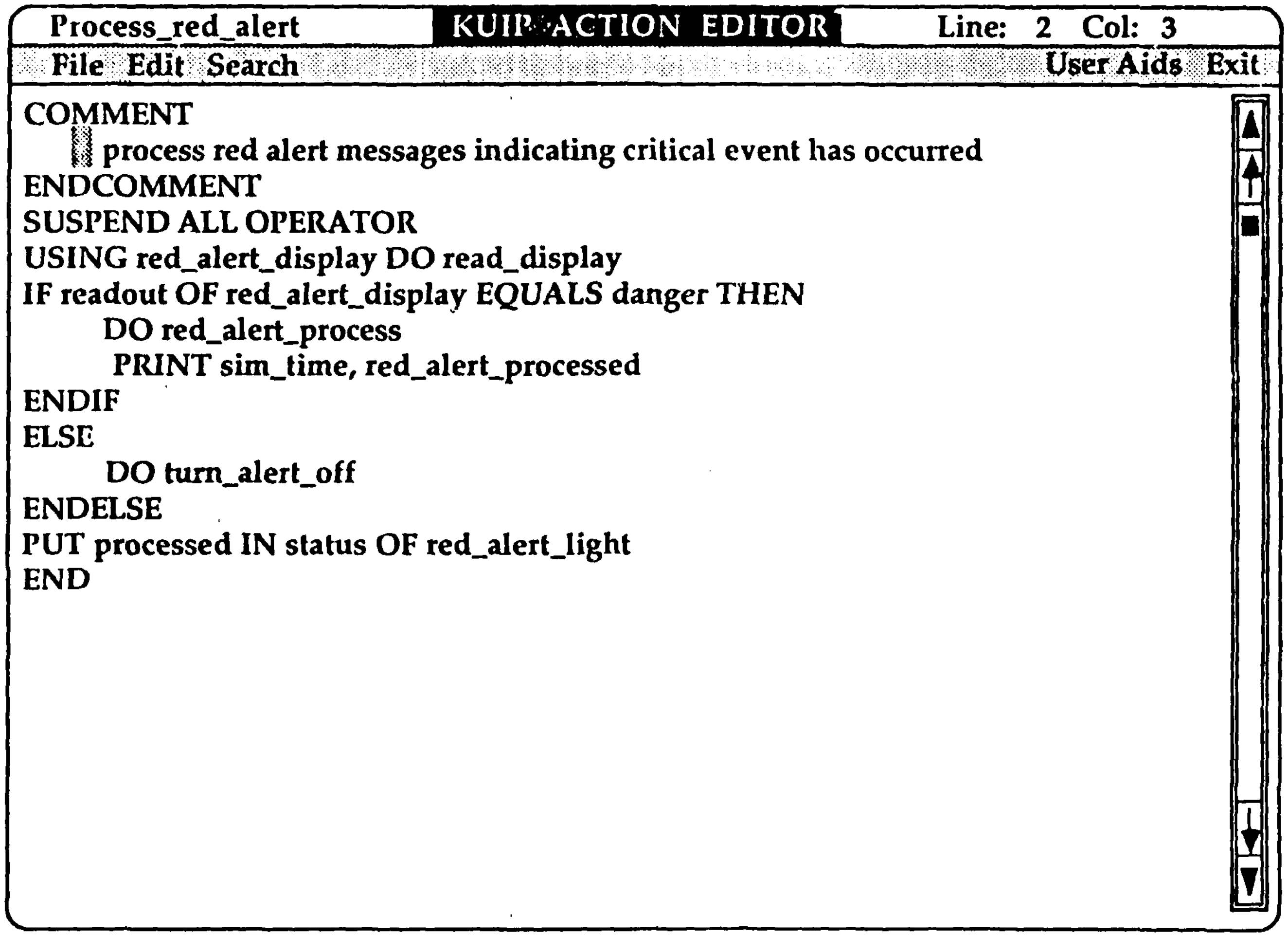
Rule Name: red_alert

If: type OF message EQUALS red_alert

Do: process_red_alert

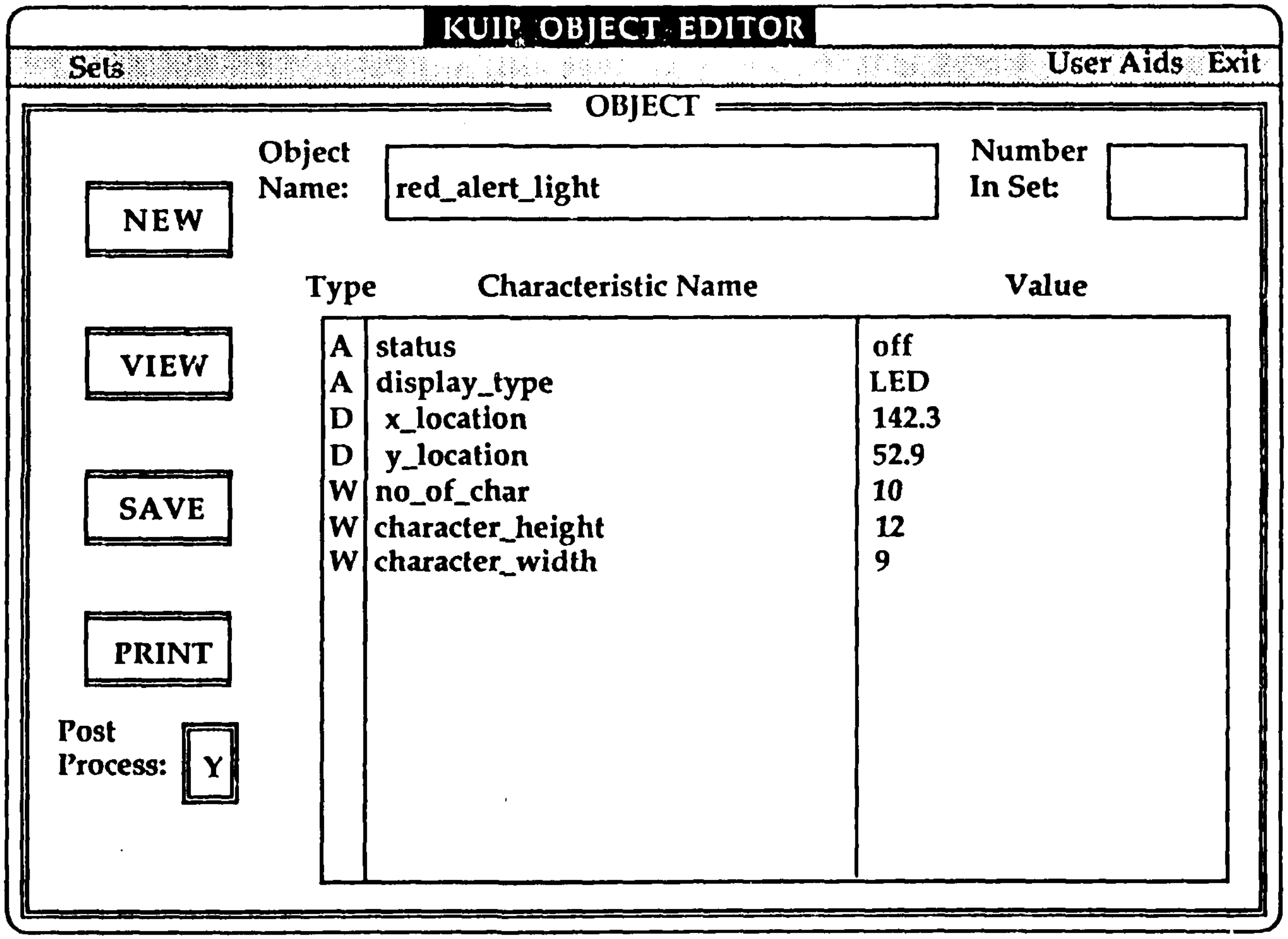
Until: status OF red_alert_light EQUALS pr

Figure 5-16: KUIP Rule Editor Screen.



```
COMMENT
    process red alert messages indicating critical event has occurred
ENDCOMMENT
SUSPEND ALL OPERATOR
USING red_alert_display DO read_display
IF readout OF red_alert_display EQUALS danger THEN
        DO red_alert_process
         PRINT sim_time, red_alert_processed
ENDIF
ELSE
        DO turn_alert_off
ENDELSE
PUT processed IN status OF red_alert_light
END
```

Figure 5-17: KUIP Action Editor Screen.



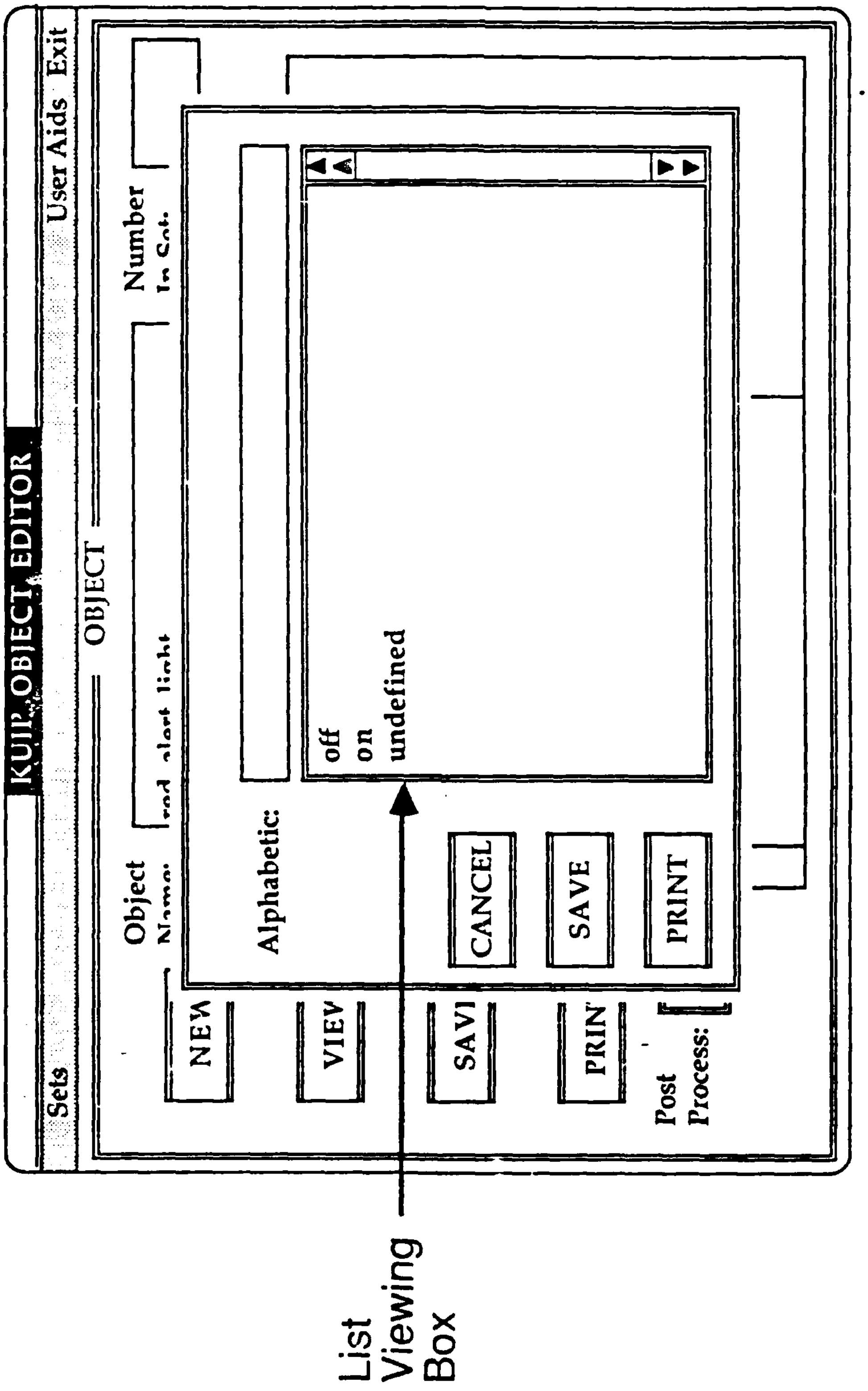
Object Name: red_alert_light

Number In Set:

Type	Characteristic Name	Value
A	status	off
A	display_type	LED
D	x_location	142.3
D	y_location	52.9
W	no_of_char	10
W	character_height	12
W	character_width	9

Post Process: Y

Figure 5-18: KUIP Object Edit Screen



off
on
undefined

Figure 5-19: KUIP Object Edit Windows.

6. DYNAMIC SIMULATION (HOS-IV)

The preliminary design and implementation of HOS-IV is being carried out under a separate contract with ARI. One part of the original effort involved designing, developing, and implementing the HOS-IV system. SEPTA will require additions and modifications to make HOS-IV usable in the SEPTA system. Many of these modifications are substantial but subtle ones involving data input, report generation, etc. In general, the philosophy behind HOS-IV remains substantially the same as previous versions. The main components of HOS-IV are the following:

- The knowledge representation scheme based upon an object-attribute structure,
- The HOS-IV simulation language,
- The human models to include the basic human performance capabilities, perception, recall, mental computation, and anatomy movement, and generate an indication of success/failure and performance time, and
- The data analysis of simulation results.

The high-level design goals of the HOS-IV system detailed in the original specifications have been implemented and include the following:

- Development of an user-oriented interface,
- Provide user-controlled level of simulation detail,
- Provide a highly flexible simulation structure that permits the user to modify/alternate human performance models, allow more direct user control over simulation sequencing, and facilitate parallel processing,
- Provide the user with control over the level of simulation detail,
- Provide a high degree of reusability, and
- Reduce the amount of time required to developa simulation model.

Fundamental concepts that underlie HOS-IV include the following:

- A single time unit can be defined for a simulation that cannot be modified within a simulation. For example, if the defined time unit is milliseconds, part of the simulation cannot be run in seconds unless a separate simulation is developed for that time interval.
- No time charges are incurred to process a step in an action or an action itself.
- An ability to prioritize actions must be provided along with the capability to group actions into priority groups.
- The rules governing the actions are to be separated by operator, system, and environment.

6.1 Dynamic Simulation Structure

6.1.1 Knowledge Representation

Knowledge representation defines all the information pertinent to the operator, system, and environment. The HOS-IV simulation requires definition of the following types of information:

- Objects
- Action Rules
- Actions
- Discrete Events

A description of each of the types of information follows.

6.1.1.1 Objects

HOS-IV uses an object-attribute structure to store data on each entity, or object, to be modeled in the simulation (e.g., displays, controls, sensors, aircraft). Each object has an associated list of attributes (e.g., size, location) and each attribute is assigned a value. These attributes describe the important features or characteristics of an object. Since the term 'attribute' was considered difficult for users to understand, the term characteristics is used instead. Values indicate the state of the characteristic at a particular point in the simulation. Objects can be defined at various levels of detail. For example, the object EMERGENCY INDICATOR LIGHT can have one characteristic, STATUS, with a value of ON or OFF. Another object may have several characteristics, such as ENEMY EMITTER with characteristics for LOCATION, SPEED, BEARING, FREQUENCY, PULSE WIDTH, PULSE REPETITION FREQUENCY, and PULSE MODE, along with their assigned values. The values of the characteristics of OBJECTs indirectly control the flow of events in the simulation by providing information to the RULES and ACTIONS.

An example of an object is as follows:

Object Name:	Radar
Characteristics:	
Status	on/off
Mode	automatic/manual
X-location	46.78
Y-location	124.54

In the example above, the user has defined the x and y crewstation coordinates of the radar control panel. The z coordinate is not important for this particular simulation so the user chcse not to define it. Here, the user is interested in determining motor activity times, that is, the time required by the operator micromodels to move their eyes and hands to the panel. Alternatively, the user may not define location and use fixed time charges for operator motor activities.

The object-characteristic structure provides a powerful and efficient knowledge representation capability that provides the user with the flexibility to define what is important to the application at any level of detail.

6.1.1.2 Action Rules

Action rules drive what actions will be executed at a particular simulation time snapshot. These rules are defined by a starting and ending conditional, the name of the action to be invoked if the starting conditional is true, and a priority assignment from 0 (lowest) to 9 (highest). Separate queues are provided for the operator, hardware, and environment. If the starting conditional statement is true, the named action will be invoked. The action will then continue until the ending conditional is true. Rules can be declared active or inactive during the simulation. At each simulation snapshot, the conditional statements of all active rules will be evaluated to determine what actions will be executed or terminated. The rules will be processed in the sequence specified by the priority assignment. This precedence relationship is particularly useful as actions invoked by a high priority rule may suspend lower priority rules. This capability permits rules to be categorized by mission critical tasks as well as mission phase. All active rules represent the processes that are occurring in parallel. If sequential processing mode is desired, the rules must be set up in such a fashion that only one rule is active at any particular time.

Each rule is defined by three components:

- An IF clause,
- A DO clause, and
- An UNTIL clause.

An example of a rule is:

IF status OF radar EQUALS on
DO process_emitters
UNTIL status OF simulation EQUALS off

This rule will be invoked only when the UNTIL clause is false and the IF clause is true.

6.1.1.3 Actions

Actions describe what will be done by the operator, system, and environment at a given simulation snapshot if the related rule is true. The steps to accomplish a task which must be performed at a given mission time based on the current environmental and system status are detailed in the action. Actions can include updates to the values of object characteristics, invocation of other actions, and the initiation or suspension of action rules. Actions are the only simulation mechanism which can affect the values of the characteristics of objects.

Actions are defined using a small set of standard verbs (e.g., PERFORM, SET, SUSPEND) known as the HOS-IV Action Language (HAL). One of the major decisions in developing HAL involved whether to incorporate a true natural language capability into HOS-IV or to utilize a limited set of verbs. An unrestricted natural language capability would provide greater freedom of expression, but for HOS-IV simulations it does not appear to be necessary and in many cases would lead to greater ambiguity. Although unrestricted natural language is the least limited to the user, it is not currently practical to machine-parse on an IBM PC/AT level computer because of its extreme complexity. Additionally, a body of evidence suggests that limited the vocabulary and syntax available to the user improves the user's ability to utilize and comprehend the language (Bailey, 1985; Hendler & Michaelis, 1983). Therefore, it was decided to develop a limited set of verbs with restricted syntax to specify procedural knowledge. For HOS-IV, a small set of verbs which convey the required actions has been developed. Verbs are the 'primitive' building blocks with which the user constructs complex and application-specific actions. A summary of the current set of HOS-IV verbs is shown below.

COMMENT <string> ENDCOMMENT

DEFINITIONS [<def-statement>] ENDDEFINITIONS

END_SIM

FILE [<print-value>] ENDFILE

GET <local> FROM <attribute> OF <object>

IF <boolean> THEN <statement-group> ENDIF {ELSE <statement-group> ENDELSE}

PRINT [<print-value>] ENDPRINT

PUT <send-value> IN <attribute> OF <object>

RETRIEVE <local-object> FROM <set-keyword> <set-name>

SET <local> TO (<formula>)

START <rule number>

STOP <rule number>

```
SUSPEND <rule number>
USING [<parameter>] DO <proc-name>
WHILE <boolean> THEN <statement-group> ENDWHILE
```

The example below illustrates the use of the SET, IF, and PUT verbs:

```
SET random_number TO (RAND()/32767.0)
IF prob_recall GREATER_OR_EQUAL random_number THEN
    PUT recalled IN status OF memory_cell
ENDIF
```

The SET verb is used to set a local variable equal to a mathematical function. The IF verb contains a conditional statement to determine which logic is to be followed next. If the statement is true, the PUT verb is used to modify the object data base.

One of the convenient features of HOS-IV is that models written in the C language can be directly integrated into the HOS-IV ACTION language. In order to embed a model of a radar processor which was written in C, the Key Word START_C_CODE is placed before the module and END_C_CODE at the end of the module. HAL provides full functionality in data manipulation and simulation processing.

6.1.1.4 Discrete Events

HOS-IV permits the definition of discrete events in order to simulate processes that are not originated by the operator but which have an impact on the course of the simulation. Discrete events are defined by the event-name, the time of the event, and the action name to be executed. Typical events can include special tasking messages communicated to the operator from a higher command, hardware system failures, or environmental changes (e.g., weather). The user defines the name of the event, time of event, and the action to be triggered. For example:

At Time:	00:05:00.000
Do:	system_failure
Event Name:	Electrical storm causes power outage.

6.1.1.5 System Interactions

Figure 6-1 illustrates the interactions and functional flow of information within HOS.

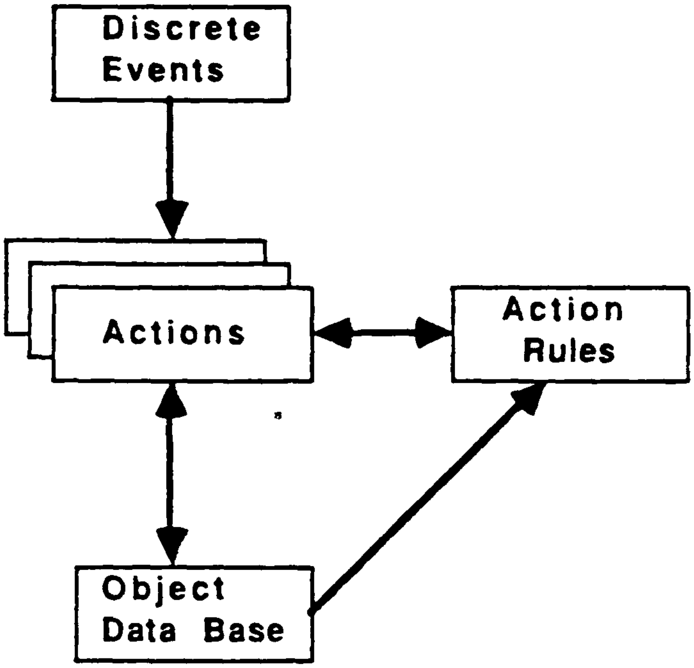


Figure 6-1. HOS-IV Simulation Structure and Functional Flow

As illustrated in Figure 6-1, OBJECTs store values for all the entities to be simulated. The user partially controls the level of simulation detail by the number of objects and the type of characteristics defined for each object. For example, enemy aircraft can be a single object with an associated probability of a particular number coming within range of a radar at each simulation snapshot. Alternatively, 50 separate OBJECTS can be defined, each representing a unique aircraft with a specific geographic location relative to the radar system. The values of objects are dynamically controlled by ACTIONS which occur during the simulation. ACTIONS are triggered by a set of RULES. RULES define the conditions which will control the flow of events in the simulation.

Depending on the state of the object data base at each simulation time snapshot, RULES will trigger ACTIONS. ACTIONS in turn can suspend or activate RULES as well as modify the data base. Typically, RULES are grouped by mission phase. If, in an action, it is determined that a new mission phase has started, the ACTION will suspend a whole set of RULES and activate other RULES. DISCRETE EVENTS can also trigger ACTIONS.

6.1.2 User-Computer Interface

The user-computer interface (UCI) is the heart of the new system and has been designed specifically for the IBM PC-AT with enhanced color graphics capabilities and a mouse pointing device. The UCI dialogue is based upon pull-down menus and input templates which assist the analyst in entering the necessary simulation information. The UCI guides the analyst at every step in the simulation process, thereby facilitating simulation definition, execution, and analysis. Not only does the UCI facilitate access to HOS, but it also aids and augments this interaction in significant ways through the incorporation of numerous user aids. These user aids include a menu-driven interface; on-line guidance via a context-sensitive HELP capability; extensive libraries of standard models, parameters, and knowledge; and detailed and easily understood error messages and recovery procedures.

6.1.3 Modeling Capability

The central models resident in HOS-IV are the operator models which will initially generate performance time and an indication of success/failure. The current version of HOS-IV contains cognitive (recall, attention, mental computations), perceptual (visual and auditory), and psychomotor (anatomy movement) models which are based on experimental data from the human performance literature.

The revised modeling capability in HOS-IV incorporates major improvements in the areas of efficiency, usability, and adaptability. HOS-IV, in particular, provides ready access to a library of standard models for items commonly included in simulations such as controls, displays, etc. In addition, the HOS-IV analyst will be able to tailor the models to the needs of the particular application by specifying the appropriate level of detail for each model (or, if desired, incorporating his or her own models). All models included in HOS-IV are written utilizing the same HAL language as used to define simulation actions. The models are described in detail in the HOS-IV User's Guide (Harris, et al., 1988).

6.1.4 HOS-IV Outputs

The HOS-IV outputs include:

- A timeline of events for the operator, system, and environment,
- User-defined measures of effectiveness, and

- Standard analysis, such as.
  - Mean time to complete an action,
  - Number of times an action is performed,
  - Proportion of the operator's time spent on each action.

HOS-IV allows for user-controlled detail of simulation outputs. When building the object library, the user can identify objects whose values are to be tracked throughout the simulation. User-defined measures of effectiveness, such as the number of contacts processed per hour or a history of error rates, can also be defined and stored in a separate simulation file at simulation end . With these data, the user can identify operator and system bottlenecks and determine periods of operator overload and the circumstances surrounding those overload periods. The models are described in detail in the HOS-IV User's Guide (Harris et al., 1988).

6.1.5 MicroModel Library

A library file will allow access to the micromodel database to facilitate finding, editing, or creating micromodels. Directly connected to this library or inventory will be a module which we will call a Bill of Behaviors (BOB) which will facilitate the user in constructing new micromodels. The way this will work will be to allow short-hand entries when constructing micromodels. The user will enter a micromodel name off the library/inventory list and the BOB module will fetch the micromodel. The user will then be able to use it as is or to modify it as appropriate. If modified, he will be able to save the module in the master micromodel file for later use in another analysis. This BOB module cannot be designed until we have greater experience with the new HOS-IV and have developed the overall structure of the human model.

Definitions:

Inventory: a list of all micromodels written and currently available

MicroModel: A detailed and generalized description of the specific behavior to be modeled; a model of one or more Underlying Internal Processes. For example, SHORT-TERM MEMORY would be a micromodel as would GRASPING. A procedure makes call to a MicroModel passing parameters about the object to be grasped.

6.1.6 HOS-IV Simulation Approach

A HOS-IV simulation can be developed dynamically using a layered approach. The user can first define the external environment (e.g., enemy aircraft, emitters, etc.). Next, the system can be defined (e.g.,sensors, processors). Then the operator system interface of displays, controls, actions can be specified followed by operator actions that describe the perceptual, motor, and cognitive activities After this, the tactics to be employed for such actions as handling high workload situations or critical events can be developed. The user can develop any one of these layers to the level of detail necessary. The user can also vary the discrete events during the simulation to analyze the effect of a single system failure followed by multiple failures within a preset period of time.

6.2. HOS-IV Description

This section presents a high-level functional description of each of the HOS-IV processes with the overall system organization described in Section 6.2.1.

6.2.1 System Flowchart

The HOS-IV software is split into the following major system modules as described below:

- **HOS-IV**: high-level system controller that manages the spawning processes for lower-level modules.
- **SELECT:** determines which of the existing simulations is to be used during the HOS-IV simulation session and permits the user to define a new simulation.
- **SETUP:** determines basic simulation parameters such as simulation name, time units, simulation start time, and maximum simulation time based upon .
- **EVE_EDIT**: maintains the event data base for a simulation including creation of new events, modification of existing events, and deletion of existing events.
- **RULEEDIT**: maintains the rule data base for a simulation creation of new rules modification of existing rules, and deletion of existing rules.
- **ACTEDIT**: maintains the action library and processes user actions to create new actions, modify existing actions, and deletes existing actions. In addition, ACTEDIT invokes the action translator (HAL) that translates the action into C code and determines if the action contains any errors.
- **EDIT_OBJ**: maintains the object and alphabetic libraries and processes user actions to create new objects/alphabetics, modify existing objects/alphabetics, and delete existing objects/alphabetics.
- **HAL**: evaluates the syntax of an action and if no errors are detected translates the action into HOS-IV C code.
- **SIMLINK**: integrates all the user's events, rules, actions, and objects into files that are compiled using the Microsoft C compiler and builds the simulation executable file.

- **SIMRUN**: executes the user's simulation and creates files for post-processing.
- **RESULTS**: generates simulation reports.

The organization of the HOS-IV software is presented in Figure 6-2 with the left to right progression showing the sequence in which the modules must be invoked in order to properly construct and execute a simulation. The SELECT module must always be selected at the beginning of each HOS-IV session in order for the user to specify which simulation is to be used during the session. The sequence of the remainder of the modules is user-driven. All of the modules use the User-Computer Interface (UCI) described in Section 4.

6.2.2.1. HOS-IV

HOS-IV is the main module in the dynamic simulation system and is invoked when the this portion is initiated. It is the top level module that controls interactions with all other modules of in the HOS-IV system. Whenever the user terminates one of the lower level modules, control is returned to HOS-IV for the selection of the next module or to return to the top-level screen.

Description

HOS-IV controls user access to all of the HOS-IV modules. There are nine commands which can be executed from the main screen of HOS-IV — *Select Simulation, Setup Simulation, Edit Events, Edit Rules, Edit Actions, Edit Objects, Create Simulation, Run Simulation,* and *View Results.* A pushbutton was created for each function and the user invokes the desired function by moving the mouse over the title of the selected pushbutton and clicking the mouse. HOS-IV automatically initiates the Select Simulation command on start up (see section 6.2.2.2). After the user has successfully completed the simulation select step, any of the other functions may be chosen.

HOS-IV Screens

The HOS-IV screen contains a diagram of pushbuttons and arrows indicating the functional flow of the steps involved in developing executing, and analyzing a HOS-IV simulation.

Action Editor Title Bar. The title bar of HOS-IV contains the words 'HOS IV' as the function name. HOS-IV does not use the current activity or status information areas of the title bar.

HOS-IV Menu Bar. HOS-IV contains the following menu options on the menu bar as illustrated in Figure 6-3:

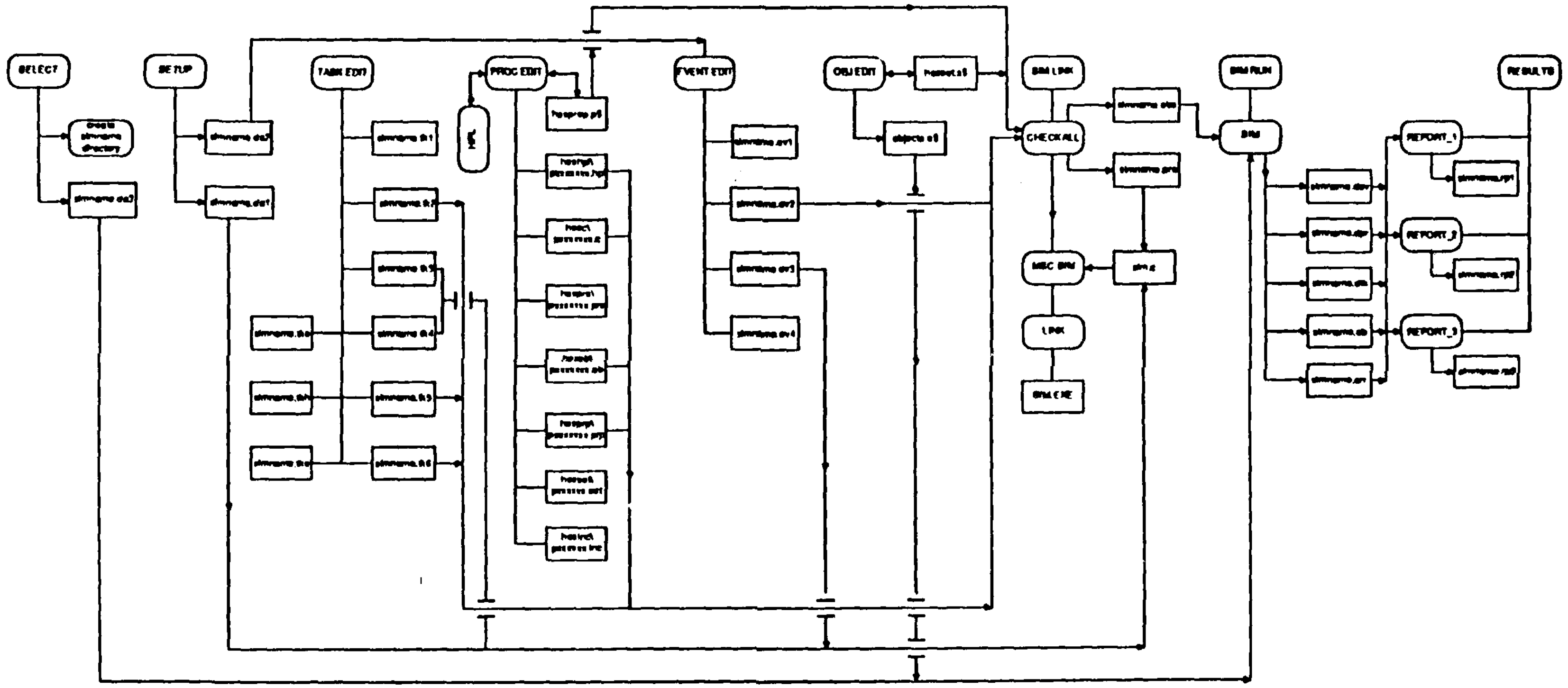


Figure 6-2. Dynamic Simulation Software Flow Chart

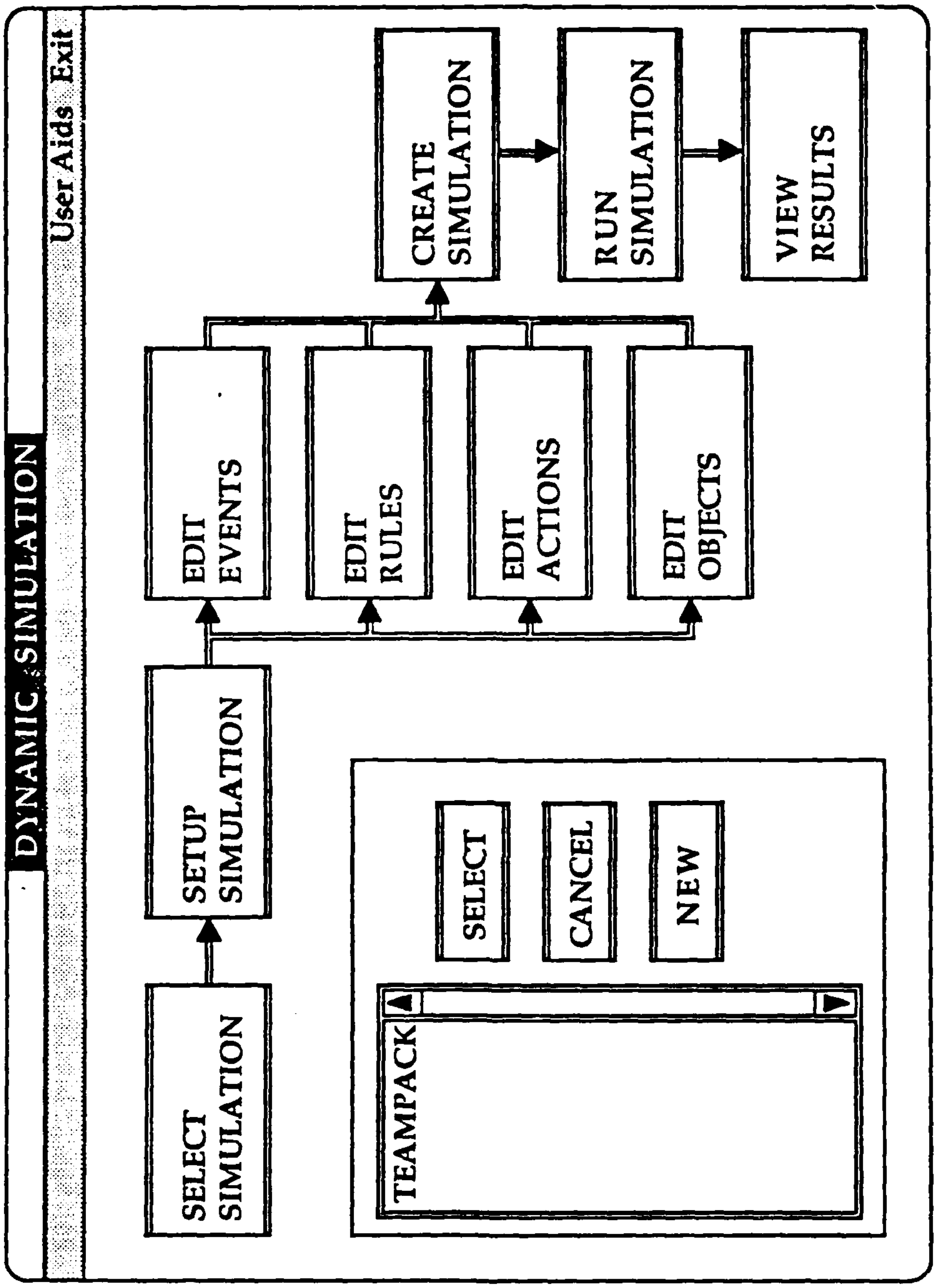


Figure 6-3: Dynamic Simulation Screen with Select Simulation Box.

- **User Aids**: provides the capabilities to view help files.
- **Exit**: terminates HOS-IV and returns the user to the main SEPTA menu.

The **User Aids** pull-down menu contains the **Help** commands that allow the user to obtain help windows about the HOS-IV module.

HOS-IV Main Screen. The HOS-IV main screen contains the following pushbuttons:

- **Select Simulation**: calls the select simulation routine
- **Setup Simulation**: spawns the setup simulation module
- **Edit Event:** spawns the event editor
- **Edit Rule**: spawns the rule editor
- **Edit Action**: spawns the action editor
- **Edit Object**: spawns the object editor
- **Create Simulation**: spawns the simlink module
- **Run Simulation**: spawns the simulation
- **View Results**: spawns the view results module

6.2.2.2 SELECT — Select Simulation

The SELECT module determines the simulation to be processed and permits the user to define new simulations.

Description

The SELECT module is automatically executed whenever HOS-IV is first executed so that the user is forced to select a simulation before any other HOS-IV functions can be invoked. In addition, the user can depress the SELECT pushbutton whenever the HOS-IV main menu is displayed in order to switch to a different simulation. If any simulations exist, SELECT displays a list selection box that allows the user to scroll through the list of currently defined simulations and select one of the listed simulations. The user can name a new simulation or cancel the select function. If no simulations exist, the user can only enter a new simulation name or cancel. Whenever the user cancels SELECT without selecting a simulation, an informative message is displayed indicating that the HOS-IV session cannot continue until a simulation is selected. The response to this message is to exit (terminate the HOS-IV session) or continue (return to SELECT). The name of a simulation can contain a maximum of eight alphanumeric characters. The simulation name cannot contain any special symbols except underscore (_). Examples of valid simulation names are my_sim, teampack, etc.

SELECT Screens

The SELECT simulation user window overlays the main HOS-IV simulation flow screen and is illustrated in Figure 6-3.

Title Bar. The title bar of SELECT contains the words 'HOS-IV' as the function name. SELECT does not use the current activity or status information areas of the title bar.

SELECT Menu Bar. SELECT contains the following menu options on the menu bar as illustrated in Figure 6-3:

- **User Aids** — provides the capability to print the simulation names and obtain help messages and;
- **Exit** — terminates SELECT and returns the user to the HOS-IV screen.

The **User Aids** pull-down menu contains commands that allows the user to print the list of simulation names and obtain help messages. It contains the following commands:

- **Print simulation names** — allows the user to obtain a printout of names of all defined simulations on the line printer.
- **Help** — allows the user to obtain additional information about using SELECT.

SELECT Windows. The main SELECT window is a dialog window for entering object information. The SELECT dialog window, as illustrated in Figure 6-3, contains the following pushbuttons:

- **SELECT** — makes the current simulation the simulation name selected in the list selection box and writes the name to the.file d:\hosiv\currsim.dat
- **CANCEL** — terminates the SELECT function and returns the user to the main HOS-IV window without changing the current simulation. If no simulation has been selected during the session, a warning message will be displayed.
- **NEW** — defines a new simulation name and generates the new simulation dialog window.

The list selection box contains the list of currently defined simulation names.

When the user selects the **NEW** button from the SELECT dialog window, an new simulation window is displayed as shown in Figure 6-3. The user can select from the following pushbuttons:

- **CANCEL** — cancels the new simulation function.
- **OKAY** — validates the entered simulation name to ensure that it does not contain any illegal characters and has not been previously defined. If the

simulation name is valid, the simulation name is added to the list and the simulation directories are created.

The alphabetic dialog window also contains a text entry box for entry of the simulation name.

SELECT Message windows. SELECT generates the following message windows.

- **No simulation selected** — informs the user that a simulation must be selected before HOS-IV can be run. The user options are to continue and select a simulation or exit from HOSIV.
- **Name already used** — informs the user that the simulation name just entered is not unique and must be re-entered. The user must hit the okay button to return to the Define new simulation window.
- **Invalid character** — informs the user that the simulation name just entered contains an invalid character and must be re-entered. The user must hit the okay button to return to the Define new simulation window.

Input/Output

The names of all existing simulations are stored in the file d:\hosiv\all_sims.dat. SELECT first determines if this file exists. If it does, the list of simulation names contained in the file are displayed in the list selection box; otherwise it prompts the user to enter the name of a new simulation. The name of the selected simulation is stored in the file d:\hosiv\currsim.dat. The contents of these files are shown below:

d:\hosiv\all_sims.dat

This file contains a list of eight character simulation names with each name in a separate record.

```
char          name          [8];
```

d:\hosiv\currsim.dat

This file contains the eight character simulation name of the current simulation.with the name stored in a separate record.

```
char          name          [8];
```

Error handling

Message windows are generated for the following conditions:

Simulation name contains invalid characters.

Simulation name has been previously defined.

A simulation must be selected before any HOS-IV processing can occur.

6.2.2.3 SETUP — Setup Simulation

The Setup Simulation module of HOS-IV maintains general information needed to run each simulation. This information includes the minimum time unit, start time, maximum simulation time, start action, and simulation description.

Description

The purpose of the Setup module is to define initialization information needed to run each simulation. To execute the Setup Simulation module, the user can depress the SETUP pushbutton from the main HOS-IV module. When the Setup Simulation module is executed, a dialog window is created with a entry fields created for each required piece of information. These fields include the minimum time unit, the simulation start and maximum times, the start action, and the simulation description. The minimum time unit can be one of seven values: days, hours, minutes, seconds, 0.1 seconds, 0.01 seconds, and 0.001 seconds. The user scrolls through a list containing these choices. The maximum simulation time and start simulation time consist of seven fields: days, hours, minutes, seconds, tenths of seconds, hundredths of seconds, and thousandths of seconds. Only those fields greater than or equal to the minimum time unit can be accessed by the user. Invalid fields are blocked out by a blue box covering the field. The user can enter any eighty character string for the *simulation description. The start action* is any valid defined HOS-IV action.

SIMSET Screens

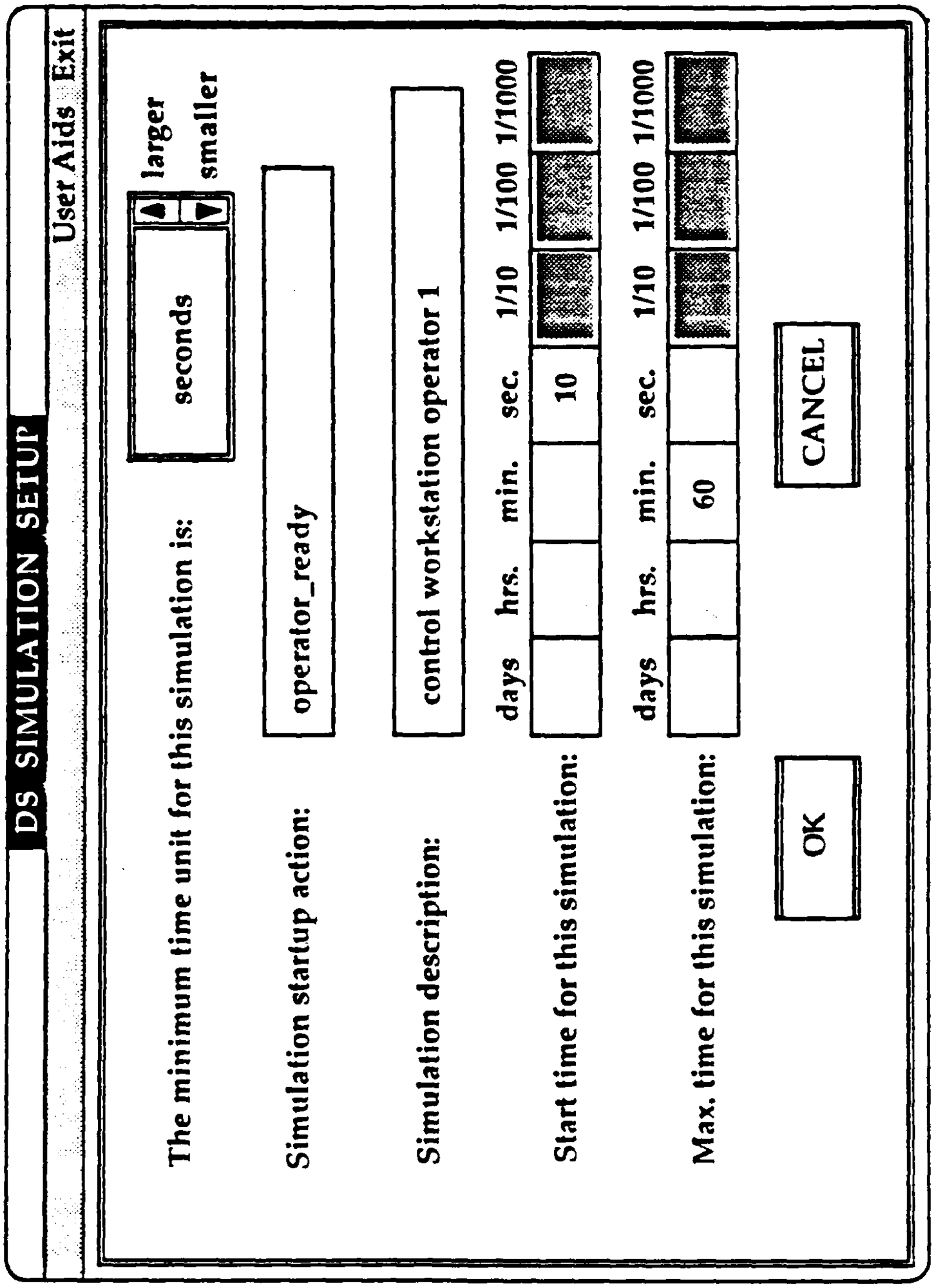
Title Bar. The title bar of SIMSET contains the words 'Setup Simulation' as the function name. SIMSET does not use the current activity or status information areas of the title bar.

SIMSET Menu Bar. SIMSET contains the following menu options on the menu bar as illustrated in Figure 6-4:

- **User Aids** — provides the capability to print the setup information, view the contents of any action file, and obtain help messages, and
- **Exit** — terminates the setup simulation module and returns the user to the HOS-IV screen.

The **User Aids** pull-down menu contains commands that allows the user to view other files, print current setup simulation information, and obtain help messages. It contains the following commands:

- **View Files** — allows the user to obtain a window that displays currently defined actions.



The minimum time unit for this simulation is: seconds

Simulation startup action: operator_ready

Simulation description: control workstation operator 1

	days	hrs.	min.	sec.	1/10	1/100	1/1000
Start time for this simulation:				10	▨	▨	▨
Max. time for this simulation:			60		▨	▨	▨

Figure 6-4 Simulation Setup

- **Print** — allows the user to obtain a printout of setup simulation information.
- **Help** — allows the user to obtain additional information about using the setup simulation module.

SIMSET Windows. The main SIMSET window is a dialog window for entering setup simulation information. The simset dialog window, as illustrated in Figure 6-4, contains the following pushbutton:

- **SAVE** — saves the current setup simulation information as displayed on the screen. The following validation is performed prior to the actual saving of the event definition:

  The start time is compared to the maximum simulation time. If the start time is greater than the maximum simulation time, an error message window is displayed indicating that the start time is too large.

  The start action is evaluated to determine if the action name is valid and has been defined. If it has not, a warning message window is displayed indicating that the action is undefined.

The simset dialog window contains the following text entry boxes:

- **Description** — entry of the simulation description as a maximum of 80 characters.
- **Startup action** — entry of an action name.
- **Maximum Simulation Time** — entry of the maximum simulation time in the form of seven two digit numbers, one for each time unit.
- **Start Simulation Time** — entry of the start simulation time in the form of seven two digit numbers, one for each time unit.

A list selection box showing the minimum time unit is displayed in the upper right corner of the simset dialog window as shown in Figure 6-4.

Input/Output

SIMSET uses the following files:

- currsim.dat — contains the current simulation name.
- simname.da1 — contains 'C' code to set the minimum time unit, maximum simulation time, start simulation time, simulation description, and start action. This file is also used as input to the Setup Simulation module to read in the current simulation information.
- simname.da2 — the first record contains an integer (0-7) to indicate the minimum time unit; the second record contains the eighty character simulation description.
- simname.da3 — the first record contains the start time in terms of the minimum time unit; the second and third records contain actual date and time that the simulation started running and stopped running. These last two records are not currently being used.

- simname.da4 — contains the start simulation action name

Error handling

When the user attempts to save the Setup Simulation information, validation checking is performed on the times and start simulation action name. An error is generated if the start time is greater than the maximum simulation time and a descriptive message is displayed in a message window indicating that 'The start time, 00:00:00:00.000, is greater than the maximum simulation time'. Errors must be corrected before a successful save can be accomplished. A warning is generated if the entered action name is undefined. The warning message window is displayed with the message, 'The action _________ has not been defined'.

6.2.2.4 EVENTEDIT — Edit Events

The EVENTEDIT module of HOS-IV maintains information on all user-defined events. Events are actions that are executed at a user defined time during the simulation to execute some time dependent occurrence. An event consists of an event description, the event action, and the event time that the event action should be executed at.

Description

EVENTEDIT maintains the list of events for the simulation, permitting the user to view, edit, delete, and create events. The data structure used to maintain the events is a linked list of event records. Each record contains the following elements:

- Event number — Used by the C code generated by HOS-IV as an index for an array of action pointers.
- Event action — A valid defined HOS-IV action name.
- Event time — Consists of a maximum of seven two character strings, one for each time unit.
- Event description — An alphanumeric string containing a maximum of 80 characters.

EVENTEDIT is executed from HOS-IV as a result of the user pressing the push button 'Edit Events'.

EVENTEDIT Screens

Title Bar. The title bar of EVENTEDIT contains the words 'Event Editor' as the function name. EVENTEDIT does not use the current activity or status information areas of the title bar.

EVENTEDIT Menu Bar EVENTEDIT contains the following menu options on the menu bar as illustrated in Figure 6-5.

- **User Aids** — provides the capability to print the list of defined events, view the contents of any action file, and obtain help messages, and
- **Exit** — terminates the event editor and returns the user to the HOS-IV screen.

The **User Aids** pull-down menu contains commands that allows the user to view other files, print all currently defined events and obtain help messages. It contains the following commands:

- **View Files** — allows the user to obtain a window that displays currently defined actions.
- **Print rules** — allows the user to obtain a printout of all defined events.
- **Help** — allows the user to obtain additional information about using the event editor.

EVENTEDIT Windows. The main EVENTEDIT window is a dialog window for entering event information. The event dialog window, as illustrated in Figure 6-5, contains the following pushbuttons:

- **NEW** — clears all input fields and places the text cursor in the event name field.
- **DELETE** — deletes the currently selected event.
- **SAVE** — saves the current event as displayed on the screen. The following validation is performed prior to the actual saving of the event definition:

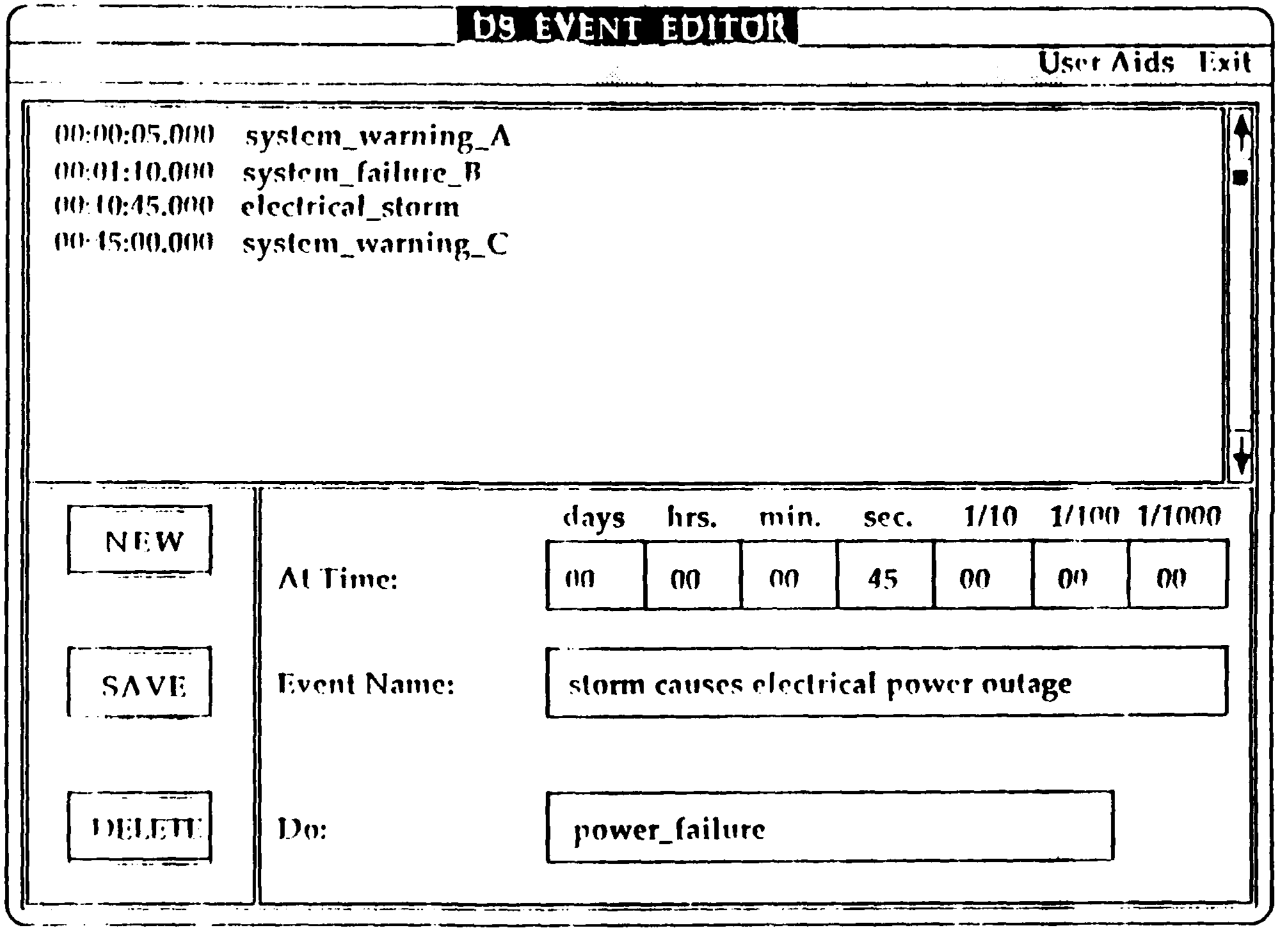
  The Event time is compared to the maximum simulation time. If the Event time is greater than the maximum simulation time, an error message window is displayed indicating that the Event time is too large.

  The Do action is evaluated to determine if the action name is valid and has been defined. If it has not, a warning message window is displayed indicating that the action is undefined.

The rule dialog window contains the following text entry boxes

- **Event Name** — entry of the Event name as a maximum of 26 characters.
- **Do** — entry of an action name
- **At Time** — entry of the time in the form of seven two digit numbers, one for each time unit.

A list selection box showing the event time, description, and event action of currently defined events is displayed in the upper left corner of the event dialog window as shown in Figure 6-5.



```
00:00:05.000   system_warning_A
00:01:10.000   system_failure_B
00:10:45.000   electrical_storm
00:45:00.000   system_warning_C
```

	days	hrs.	min.	sec.	1/10	1/100	1/1000
At Time:	00	00	00	45	00	00	00

Event Name: storm causes electrical power outage

Do: power_failure

Figure 6-5: Event Edit Screen.

Input/Output

EVENTEDIT uses the following files:

- **simname.ev1** — used by EVENTEDIT to read in the current defined events. This file is also output when the user quits. Each record contains four fields separated by a space A line is in the following format:
  - three-character event number,
  - six-character event time,
  - thirty-one-character event action, and
  - eighty-character description.
- **simname.ev3** — output by EVENTEDIT It contains the C code for the array of event action pointers. Each record contains a line in the following format:
  - event_proc[xxx] = zzzzzzzzz;
  - xxx is a number from 0 to number of events - 1. zzzzzzzzz is the event action name.
- **simname.ev2** — output by EVENTEDIT. It contains a record for each event consisting of only the event action name.

Error handling

When the user attempts to save an event, error checking is performed on the event time and event action name entered. If the event time is greater than the maximum simulation time, a message window is generated that displays the statement, "The event time, 00:00:00:00.000 is greater than the maximum simulation time." Errors must be corrected before a successful save can be accomplished.

If an action name is undefined, a warning message window is displayed with the message, "The action _________ has not been defined."

6.2.2.5 RULEEDIT — Edit Rules

The RULEEDIT module of HOS-IV maintains information for all user-defined simulation rules. Rules set up conditions that determine which actions will be executed at a particular simulation time snapshot. A rule is defined by a starting and ending conditional, the name of the action to be invoked if the starting conditional is true, and a unique task number. If the starting conditional statement is true. the named action will be invoked The action will then continue until the ending conditional is true. Rules can be of three distinct types **soldier**, hardware, or **environment** Each type of rule is grouped separately but identical code is used to maintain the rules.

Description

RULEEDIT maintains the list of defined rules for a simulation and permits the user to view, edit, delete, and create rules. There are three sets for rules for each simulation: Hardware, Soldier, and Environment. The same information is required for each rule type. A rule consists of the following elements:

Rule Number:

- Soldier — a unique three digit number consisting of a one digit priority assignment from 0 (lowest) to 9 (highest) and a two digit number 00 (lowest) to 99 (highest).
- Hardware/Environment — a two digit number 00 (lowest) to 99(highest).

Rule Name: an alphanumeric name containing a maximum of 26 characters.

If Clause: a starting condition consisting of a boolean statement utilizing characteristics of objects, constants, and properties.

Do Clause: the name of an action to be invoked when the If Clause is true.

Until Clause: an ending condition consisting of a boolean statement utilizing characteristics of objects, constants, and properties.

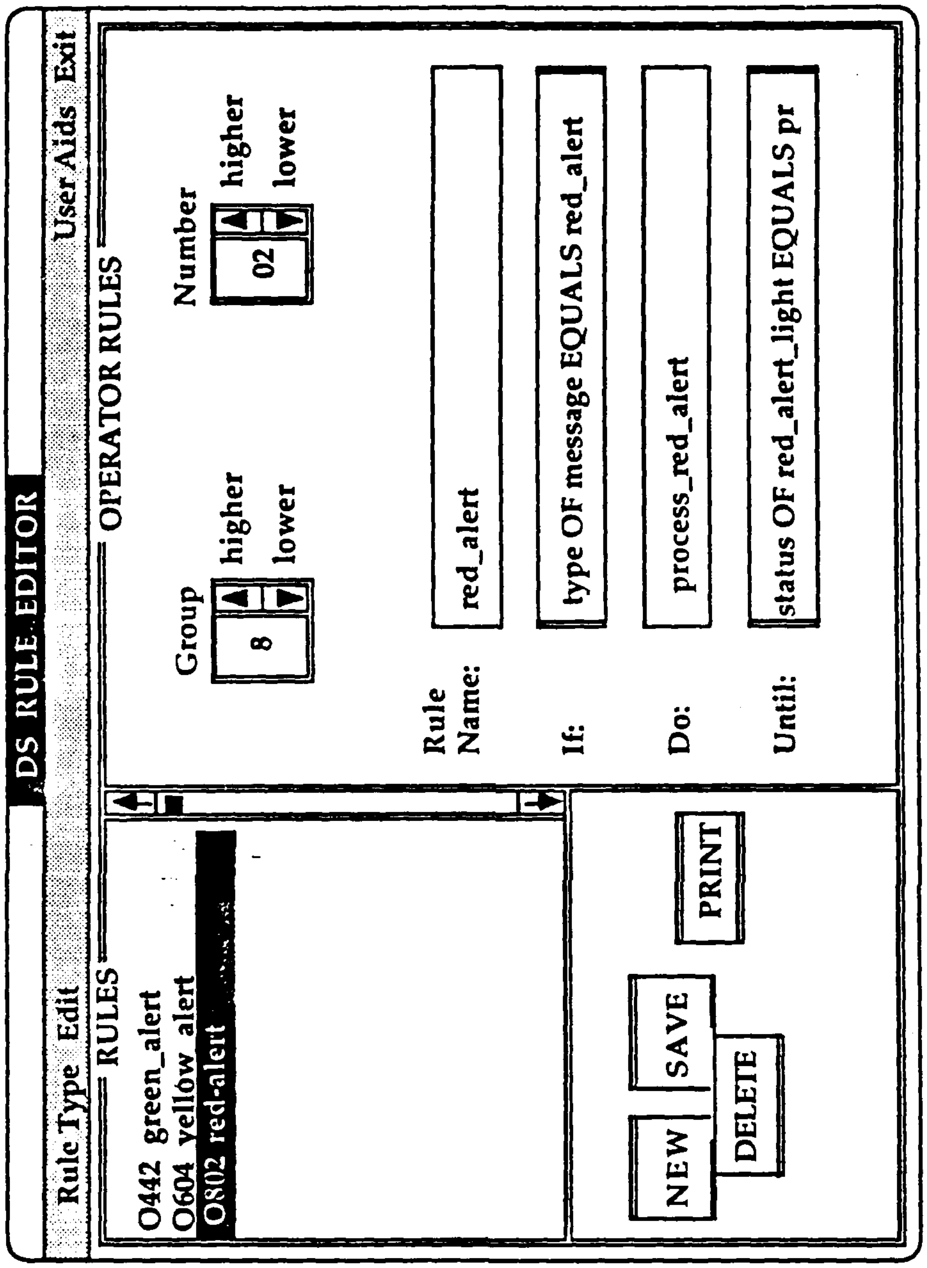
The same code performs operations on all three rule types. In order to provide the combined files needed by other modules, the rule editor combines the hardware, soldier and environment rule files upon exiting.

RULEEDIT Screens

Title Bar. The title bar of RULEEDIT contains the words 'RULE EDITOR' as the function name. RULEEDIT does not use the current activity or status information areas of the title bar.

RULEEDIT Menu Bar. RULEEDIT contains the following menu options on the menu bar as illustrated in Figure 6-[illegible].

- **Rule Types** — defines the rule type as environment, hardware, or soldier. When one of the type options is selected, the current rules, if any, are saved and the rules associated with the selected type are loaded.
- **Edit** — provides copy and paste functions. **Copy** copies the current rule into an invisible clipboard buffer. **Paste** copies the clipboard buffer into the rule currently defined on the screen only if the current rule is new.
- **User Aids** — provides the capability to print the list of defined rules, view the contents of any action file, and obtain help messages, and
- **Exit** — terminates the object editor and returns the user to the HOS-IV screen.



O442 green_alert
O604 yellow_alert
O802 red-alert

Group: 8 higher lower

Number: 02 higher lower

Rule Name: red_alert

If: type OF message EQUALS red_alert

Do: process_red_alert

Until: status OF red_alert_light EQUALS pr

Figure 6-6: Rule Edit Screen

The **rule** pull-down menu contains commands that allows the user to define the rule type. It contains the following commands:

- **Environment** — defines an environment type rule. All of the data entry fields in the rule dialogue window are blanked and the text cursor is placed in the first character in the rule name field. The rule dialog window is modified to remove the Rule Group click box if present.
- **Hardware** — *defines a hardware type rule.* All of the data entry fields in the rule dialogue window are blanked and the text cursor is placed in the first character in the rule name field.The rule dialog window is modified to remove the Rule Group click box if present.
- **Soldier** — defines an soldier type rule. All of the data entry fields in the rule dialogue window are blanked and the text cursor is placed in the first character in the rule name field.The *rule dialog window* is *modified* to add the Rule Group click box if not present.

The **User Aids** pull-down menu contains commands that allows the user to view other files, print all currently defined tasks, and obtain help messages. It contains the following commands:

- **View Files** — allows the user to obtain a window that displays currently defined actions.
- **Print rules** — allows the user to obtain a printout of all defined rules. It first formats the rules and then prints it out on the line printer.
- **Help** — allows the user to obtain additional information about using the rule editor.

RULEEDIT Windows. The main RULEEDIT window is a dialog window for entering rule information. The rule dialog window, as illustrated in Figure 6-6, contains the following pushbuttons:

- **NEW** — clears all input fields and places the text cursor in the rule name field.
- **DELETE** — deletes the currently selected rule.
- **SAVE** — saves the current rule as displayed on the screen. The following validation is performed prior to the actual saving of the rule definition:

  The rule name must be a valid name, i.e., names start with an alphabetic character (a-z), do not contain an illegal character, and is unique.

  The If and Until clauses contain valid Boolean conditions. If the Boolean operator or constant is invalid, an error message window is displayed indicating that the clause cannot be saved. If a characteristic-object pair is invalid, a warning message window is displayed indicating that there is a problem with the Boolean conditions.

  The Do action is evaluated to determine if the action name is valid and has been defined. If it has not, a warning message window is displayed indicating that the action is undefined.
- **PRINT** — prints the current rule definition on the printer.

The rule dialog window contains the following text entry boxes:

- **Rule Name** — entry of the rule name as a maximum of 28 characters.
- **If** — entry of the if clause in the form value Boolean operator value where value can be a characteristic of an object or a constant and Boolean operator can be equals, not_equal, less_than, less_or_equal, greater_than, or greater_or_equal.
- **Do** — entry of an action name.
- **Until** — entry of the until clause in the form value Boolean operator value where value can be a characteristic of an object or a constant and Boolean operator can be equals, not_equal, less_than, less_or_equal, greater_than, or greater_or_equal.

The rule dialog window contains a click box for entry of the rule group number 0-9 (soldier rules only) and rule number 0-99. A list selection box showing the number and names of currently defined rules is displayed in the upper left corner of the rule dialog window as shown in Figure 6-6.

Input/Output

The files produced by RULEEDIT are described below:

- simname.tko — stores the internal representation of the **soldier** rules and uses the following record structure for each rule:

```
char   description    [35];
char   if_cond        [200];
char   if_c    [300];
char   procedure      [35];
char   until_cond     [200];
char   until_c        [300];
int    priority;
int    sub_priority;
task_type     *next;
task_type     *last;
```

- simname.tkh — stores the internal representation of the **hardware** rules. It uses the same structure as soldier rules.
- simname.tke — stores the internal representation of the **environment** rules. It uses the same as soldier rules.
- simname.toi — contains C code to initialize the soldier tasks.
- simname.thi — contains C code to initialize the hardware tasks.

- simname.tei — contains C code to initialize the environment tasks.
- simname.toc — contains C code to execute the soldier tasks.
- simname.thc — contains C code to execute the hardware tasks.
- simname.tec — contains C code to execute the environment tasks.
- simname.to1 — contains the text version of the soldier rules. It is formatted for output to the printer including form feeds.
- simname.th1 — contains the text version of the hardware rules. It is formatted for output to the printer including form feeds.
- simname.te1 — contains the text version of the environment rules. It is formatted for output to the printer including form feeds.
- simname.to2 — contains a list of the procedures referenced in the soldier rules, each stored in a separate record.
- simname.th2 — contains a list of the procedures referenced in the hardware rules, each stored in a separate record.
- simname.te2 — contains a list of the procedures referenced in the environment rules, each stored in a separate record.
- simname.to5 — contains a list of characteristic object pairs referenced in the soldier rules. The characteristic names are separated from the object by a comma and a space and each pair is stored in a separate record.
- simname.th5 — contains a list of characteristic object pairs referenced in the hardware rules. The characteristic names are separated from the object by a comma and a space and each pair is stored in a separate record.
- simname.te5 — contains a list of characteristic object pairs referenced in the environment rules. The characteristic names are separated from the object by a comma and a space and each pair is stored in a separate record.
- simname.to6 — contains a list of all alphabetics referenced in the soldier rules, each alphabetic is stored in a separate record.
- simname.th6 — contains a list of all alphabetics referenced in the hardware rules, each alphabetic is stored in a separate record.
- simname.te6 — contains a list of all alphabetics referenced in the environment rules, each alphabetic is stored in a separate record, each alphabetic is stored in a separate record.
- simname.tk2 — contains a list of all procedures referenced in all of the rules. It is a combination of the three files: d:\hosiv\•simname.to2, d:\hosiv\•simname.th2, and d:\hosiv\•simname.te2.
- simname.tk3 — contains the C code to initialize all of the rules. It is a combination of the three files: d:\hosiv\•simname.to3, d:\hosiv\•simname.th3, and d:\hosiv\•simname.te3.
- simname.tk4 — contains the C code to drive all of the rules. It is a combination of the three files: d:\hosiv\•simname.to4, d:\hosiv\•simname.th4, and d:\hosiv\•simname.te4.
- simname.tk5 — contains the list of all characteristic object pairs referenced in the rules. It is a combination of the three files: C:\hosiv\•simname.to5, C:\hosiv\•simname.th5, and C:\hosiv\•simname.te5.

- simname.tk6 — contains the list of all alphabetics referenced in the rules. It is a combination of the three files: d:\hosiv\•simname.to6, d:\hosiv\•simname.th6, and d:\hosiv\•simname.te6.

Error handling

When the user completes entering a task and depresses the save pushbutton to save the rule, the entered fields are validated. Error messages indicate errors that must be corrected before the rule can be saved; warning messages are informative messages indicating that something suspect is contained in the rule but it does not have to be corrected before the rule can be saved. The following error messages may be generated for If and Until statements:

A piece of the statement is missing,
Syntax error: (offending token),
Undefined alphabetic or syntax error,
Syntax error, or
A task with that number already exists.

Warning messages appear when undefined entities (object, characteristic, or action) are used in the rule definition. The window indicates where the problem is, what type of entity is undefined, and what name the user entered. Since the undefined entry may be the name of an object, characteristic, or action that has yet to be defined by the user, only a warning message is generated.

6.2.2.6 EDIT_ACT — Action Editor

The action editor is used to enter actions. Actions describe what will be done by the soldier, system, and environment at a given simulation snapshot if the related rule is true. The steps to accomplish a task which must be performed at a given mission time based on the current environmental and system status are detailed in the action. Actions can include updates to the values of object characteristics, invocation of other actions, and the initiation or suspension of action rules. Actions are the only simulation mechanism which can affect the values of the characteristics of objects.

Actions are defined using a small set of standard verbs (e.g., PERFORM, SET, SUSPEND) known as the HOS-IV Action Language (HAL). A summary of the current set of HOS-IV verbs is shown below:

```
COMMENT <string> ENDCOMMENT
DEFINITIONS [<def-statement>] ENDDEFINITIONS
END_SIM
```

FILE [<print-value>] ENDFILE

GET <local> FROM <attribute> OF <object>

IF <boolean> THEN <statement-group> ENDIF {ELSE <statement-group>
ENDELSE}

PRINT [<print-value>] ENDPRINT

PUT <send-value> IN <attribute> OF <object>

RETRIEVE <local-object> FROM <set-keyword> <set-name>

SET <local> TO (<formula>)

START <rule number>

STOP <rule number>

SUSPEND <rule number>

USING [<parameter>] DO <proc-name>

WHILE <boolean> THEN <statement-group> ENDWHILE

Description

EDIT_ACT is essentially a free format word processor with word wrapping, cut and paste features, and mouse and keypad control of the text cursor. Once an action is entered, the user can specify that it be translated by HAL for use in the simulation executable. The status of the translation, whether successful or errors were detected, are displayed in a message window. If any errors were detected in the translator, the View File option can be used to create a separate window containing the translator output and accompanying error message.

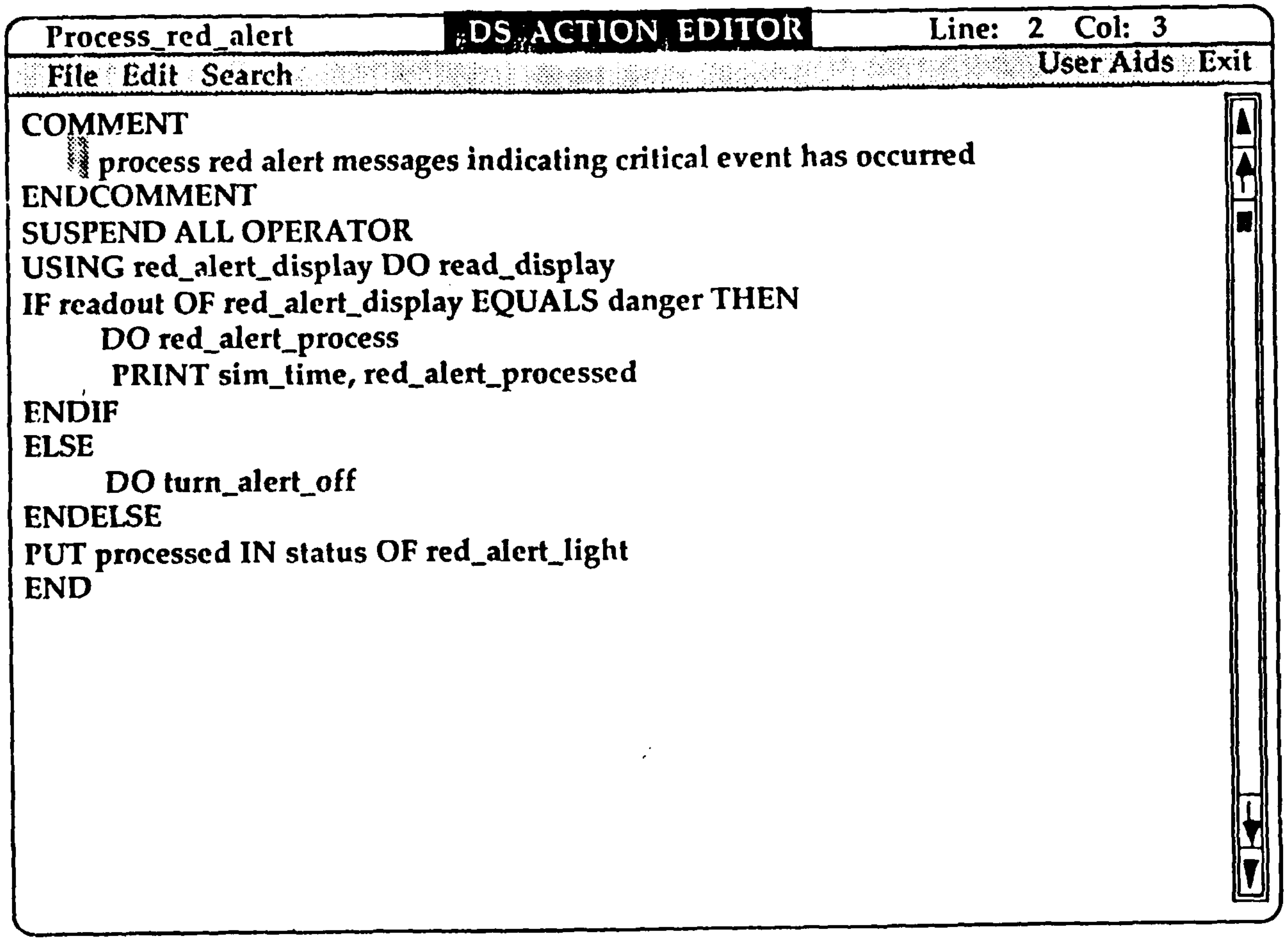
EDIT_ACT Screens

The action editor consists of a title bar, a menu bar, a text editing window with a scrollbar, and a number of dialog boxes used for program interaction with the user. The action editor screen is illustrated in Figure 6-7.

Action Editor Title Bar. The Action Editor title bar consists of a current activity area on the left which displays the name of the current action, the words 'ACTION EDITOR' in the center, and the current line and column position of the text cursor on the right.

Action Editor Menu Bar. The menu bar for the Action Editor contains the following menu options:

- **File** — file related commands such as saving. opening, etc.
- **Edit** — editing commanos, cut, paste, etc.
- ***Search*** — *word and line search commands.*



```
COMMENT
    process red alert messages indicating critical event has occurred
ENDCOMMENT
SUSPEND ALL OPERATOR
USING red_alert_display DO read_display
IF readout OF red_alert_display EQUALS danger THEN
        DO red_alert_process
         PRINT sim_time, red_alert_processed
ENDIF
ELSE
        DO turn_alert_off
ENDELSE
PUT processed IN status OF red_alert_light
END
```

Figure 6-7: Edit Action Screen.

- **User Aids** — provides the capabilities to view help files and action files.
- **Exit** — terminates Action Editor and returns the user to the HOS-IV module.

The **File** pull down menu contains:

- **New** — closes the current document after asking if changes should be saved and then creates a new empty action.
- **Open** — closes the current document after asking if changes should be saved and then opens the file selection dialog box.
- **Translate** — asks if changes to the current document should be saved and then spawns the HPL translator.
- **Save** — opens the save dialog box.

The **Edit** pull down menu contains:

- **Cut** — removes selected text from the current document and stores it in the clipboard.
- **Copy** — copies selected text from the current document and stores it in the clipboard.
- **Paste** — inserts the contents of the clipboard in the current document at the insertion point.
- **Clear** — deletes the selected text from the current document and throws it away.

The **Search** pull down menu contains:

- **Find** — opens the input search string dialog box for the user to define a search string.
- **Find Next** — finds the next occurrence of the string defined in the input search string dialog box.
- **Goto Line** — opens the input line number dialog box for the user to enter the number of the line to goto.

The **User Aids** pull down menu contains commands that allow the user receive help on the current module and view action files:

- **Help** — allows the user to obtain additional information about using ACTION EDITOR.
- **Print** — allows the user to get a formatted hardcopy on their standard printer.
- **View File** — allows the user to view action files created using the action editor.

Action Editor Text Editing Window. The text editing window consists of text area and a scrollbar. The text area is 23 rows by 77 columns. The scrollbar is to the right of the text area and has four control buttons and a relative file position indicator. The four controls on the scrollbar are:

- **Scroll Line Down** — displays a page of text starting from the line before the current top line.
- **Scroll Page Down** — displays a page of text starting one page before the current top line.
- **Scroll Page Up** — displays a page of text starting from the current bottom line.
- **Scroll Line Up** — displays a page of text starting from the line after the current top line.

Action Editor Dialog Boxes. Dialog boxes contain text entry fields in addition to pushbuttons, list boxes, and other message window components.

- **File name** — user can edit current action name and choose to save or cancel the save operation.
- **Search string** — user can input string to search for and choose to search or cancel. The search string can contain only alphanumeric characters and underscore (_). It cannot search for control characters such as tabs.
- **Line number** — user can enter integer line number only and choose to move the text cursor to that line or cancel the goto operation.

Action Editor Message Windows. Message windows display information which the user must acknowledge by hitting a pushbutton. Some message windows may be cancelled.

- **String not found** — the search string was not found in the document.
- **Translation successful** — the action was successfully translated.
- **Translation unsuccessful** — the action did not translate.
- **End editing session** — the user may confirm or cancel his Quit selection from the Exit menu.
- **Save changes** — *the user may confirm or cancel* the saving of the current action.
- **Out of memory** — the document is too large (actually if the file is this big the compiler won't accept it anyway).
- **File selection** — the user can select an action to open or delete or choose to *cancel his selection.*
- **Confirm delete** — the user can confirm or cancel his decision to delete an action.
- **Can't delete current file**— the user attempted to delete the currently opened file.
- **Can't delete system file** — the user attempted to delete the TRANSLATOR OUTPUT file or the SYMBOL TABLE file.

Action Editor Information Windows. Information windows require no input from the user. They contain informative messages to let the user know that a selected command is being processed.

- ***Printing In progress*** — the printing of an action is underway.
- **Reading file** — the file is being read.
- **Translating** — the HAL translator is running.

Input/Output

Files. EDIT_ACT references the following files:

- **d:\sortproc.p$** — sorted list of action names.
- ***d:\htemp.txt*** — *temporary file created* for saving and printing purposes.
- **c:\hoslv\hoshpl\pnnnnnnn.hpl** — actual action files where *nnnnnnn* is a seven digit sequential number assigned by the EDIT_ACT to identify the file uniquely. When the translated version of the action is stored in the p*nnnnnnn.c* file the seven digit identifier is used.
- **c:\hoslv\hoshelp\prc_help.000** — list of help topics specific to the Action Editor.
- **c:\hoslv\hoshelp\prc_help.nnn** — help files (numbered extension starting from 001).

User Actions. EDIT_ACT uses the following function keys:

- **F1** - Begin Mark — sets beginning of text selection
- **F2** - End Mark — sets end of text selection
- **F3** - Cut — *see section* 6.2.2.6.1.2 under the **Edit** menu
- **F4** - Copy — see section 6.2.2.6.1.2 under the **Edit** menu
- **F5** - Paste — see section 6.2.2.6.1.2 under the **Edit** menu
- **F6** - Clear — see section 6.2.2.6.1.2 under the **Edit** menu
- **F7** - **F10** — Not implemented

Error handling

Errors generated by the HAL translator are available to the user in a separate view window. An error flag is passed from HAL to indicate whether or not any errors were detected in the translation process. TRANSLATOR_OUTPUT contains the translated action and descriptive errors messages.

6.2.2.7 EDIT_OBJ — Object Editor

All knowledge about entities to be modeled in a simulation (e.g., displays, controls) are defined as objects. HOS-IV utilizes an object-attribute structure to manage the object data. Each object has an associated list of attributes (e.g., size, location) and each attribute is assigned a value. These attributes describe the important features or characteristics of an object. In order to enhance the user's understanding of this structure, attributes are referred to as characteristics in HOS. Values indicate the state of the characteristic at a particular point in the simulation.

Objects are stored in a library that is accessible to/from all simulations developed on a particular microcomputer. This object library provides a common facility for storing object knowledge and sharing object definitions between simulations. Whenever an object is used in a rule or action, the current object definition is retrieved from the object library. It is important to note that the object library can be shared by multiple simulations and is **not** simulation dependent. This section describes the EDIT_OBJ module of HOS-IV which maintains the object and alphabetic library.

Description

EDIT_OBJ processes all user actions related to object definitions and their associated characteristics and values. Characteristics types are **whole**, **decimal**, or **alphabetic**. Whole and decimal represent numeric values. Alphabetics are text strings and the list of defined alphabetics are stored in a separate alphabetics dictionary. EDIT_OBJ maintains the object and set libraries and the alphabetics dictionary. An object definition consists of the following information:

- **Object Name** — an unique identifier of up to 28 characters. The first character must be an alphabetic (a-z) and the remainder can contain alphabetics (a-z), numbers (0-9), and underscore (_); and
- **Characteristic List** — list of characteristic names with associated type and initial value. A maximum of 15 characteristics can be defined for an object. The characteristic name can be a maximum of 28 characters including alphabetics (a-z), numbers (0-9), and underscore (_).

The characteristic type is automatically assigned based upon the contents of the initial value. Type *whole* represents an integer number containing only the digits 0-9 and an optional preceding plus (+) or minus (-) sign. Whole values are saved as C type *long* that are stored in 4 bytes. The valid range of values for wholes is -2,147,483,648 to 2,147,483,647. Type *decimal* represents a decimal number that contains only the digits 0-9, a single decimal point (.), and an optional preceding plus (+) or minus (-) sign. Decimal values are saved as C type *double* that are stored in 8 bytes. The valid range of values for decimals is approximately 1.7E-308 to 1.7E+308. If the value is neither whole or decimal, then EDIT_OBJ assumes that it is an alphabetic value. Alphabetic values are strings of up to 28 characters that can contain any symbol except space, single quote, or double quote. The strings are entered without any special enclosing characters such as quotes. The alphabetic values are stored in an separate alphabetic dictionary.

Objects can either be defined as:

- Simple, singular objects such as a single display; or
- Sets of objects which represent multiple occurrences of identically defined objects such as a list of 10 messages or a set of 500 emitters.

For object sets, the characteristic list must be identical but the value of a characteristic can vary. The names of the members of objects in object sets are constructed by EDIT_OBJ appending a sequential member number, starting with one, to the end of the object set name. For example, the object set name EMITTER would contain objects named EMITTER001, EMITTER002, EMITTER003, etc. Once a set has been defined, the number of members in the set cannot be changed, i.e., no objects can be added or deleted from the set nor can any characteristics be modified without resaving the entire set. The only item that can be modified by the user is the value of a characteristic. When sets are initially created, all members of the set have the initial value assigned to the set name.

The object library and alphabetic dictionary are accessed by other HOS-IV modules to check for the existence of an object, object/characteristic pair, or an alphabetic value.

EDIT_OBJ Screens

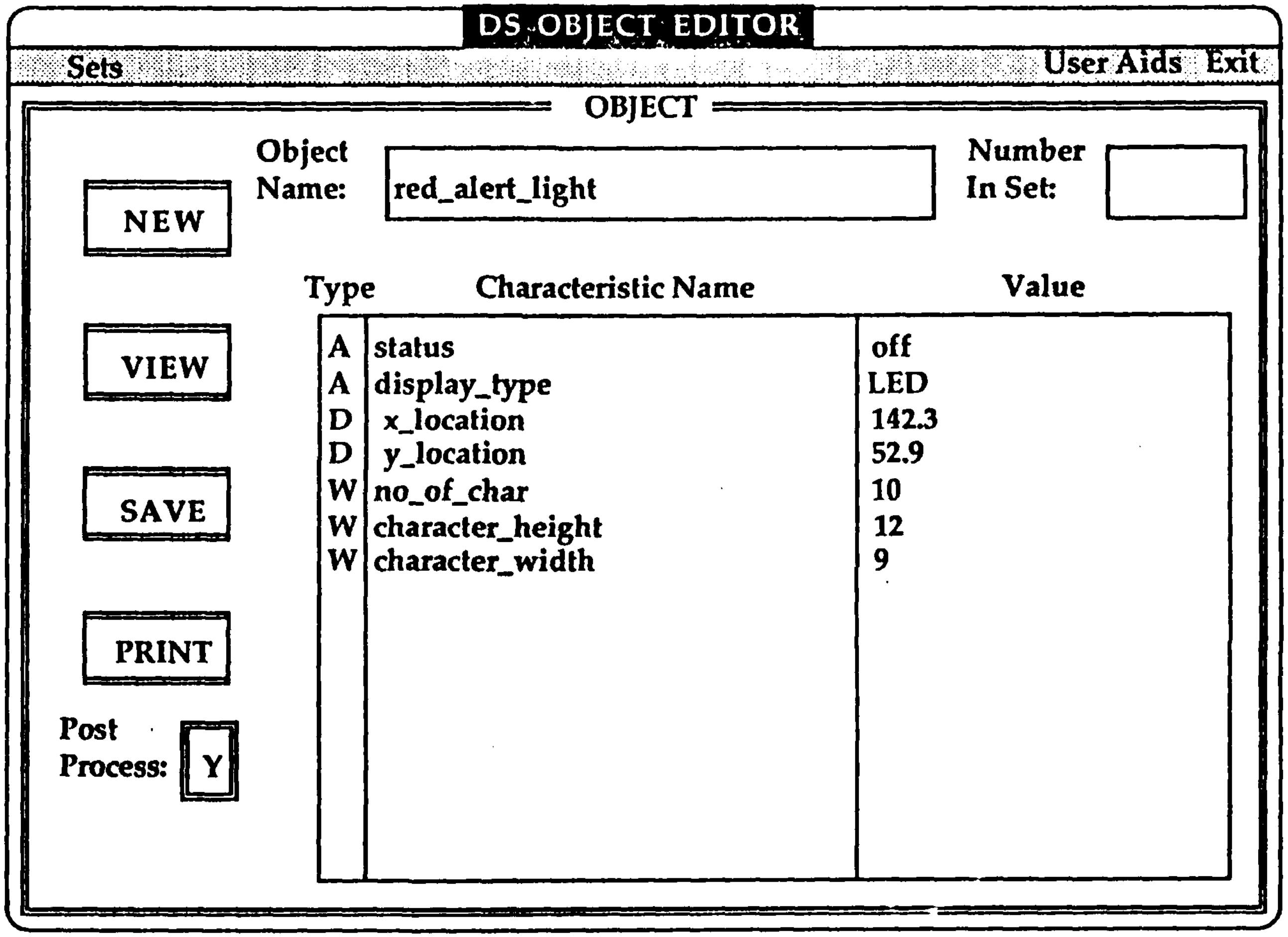
Title Bar. The title bar of EDIT_OBJ contains the words 'OBJECT EDITOR' as the function name. EDIT_OBJ does not use the current activity or status information areas of the title bar.

EDIT_OBJ Menu Bar. EDIT_OBJ contains the following menu options on the menu bar as illustrated in Figure 6-8:

- **Sets** — defines and manipulates object sets,
- **User Aids** — provides the capability to add a value to the alphabetic dictionary, print the object library, and obtain help messages, and
- **Exit** — terminates the object editor and returns the user to the HOS-IV screen.

The **set** pull-down menu contains commands that allows the user to create, save, and view object sets. It contains the following commands:

- **New set** — defines a new object set. All of the data entry fields in the object dialogue window are blanked and the text cursor is placed in the first character in the object name field.
- **Save a set** — saves an object set and creates n objects in the set where n is the number of items in the set specified in the number field. An informative message window is displayed showing the object number being created and the number in the set.



Object Name: red_alert_light

Number In Set:

Type	Characteristic Name	Value
A	status	off
A	display_type	LED
D	x_location	142.3
D	y_location	52.9
W	no_of_char	10
W	character_height	12
W	character_width	9

Post Process: Y

Figure 6-8: Edit Object.

- **View a set**— creates a list box containing the name of all currently defined sets and permits the user to select a set by pointing to the desired set name. The user has the option to either (1) delete all members in the set and the object set itself by depressing the **Delete** pushbutton, (2) open the set by depressing the **Open** pushbutton, or (3) **Cancel** the view operation. For the open set option, the contents of the first member of the set (i.e., object number 1 in the set) is displayed in the object dialogue window.

The **User Aids** pull-down menu contains commands that allows the user to create, save, and view object sets. It contains the following commands:

- **Add an alphabetic** — allows the user add an alphabetic to the alphabetic dictionary. New alphabetics are always added at the end of the dictionary because the alphabetics are implemented by using their position into the dictionary as the value to be placed in the object. However, the alphabetics are presented to the user in alphabetic order.
- **Print the object library** — allows the user to obtain a printout of all objects in the object library. It first formats the object library,and then prints it out on the line printer.
- **Help** — allows the user to obtain additional information about using the object editor.

EDIT_OBJ Windows. The main EDIT_OBJ window is a dialog window for entering object information. The object dialog window, as illustrated in Figure 6-9, contains the following pushbuttons:

- **NEW** — clears all input fields and places the text cursor in the object name field.
- **VIEW** — displays a list selection box containing the names of the currently defined objects in the object library (in alphabetic order), including members of set objects. The user can then use the point and click selection method to select an object name. The currently selected object will be highlighted in black. The following pushbuttons are functional in the view window:

  - **OPEN** — closes the view window and displays the current definition of the selected object in the object dialog window.
  - **DELETE** — deletes the object currently selected object. If the object is a member of a set, a message window is created to inform the user that *members of sets cannot be deleted.*
- **SAVE** — saves the current object as displayed on the screen. The following validation is performed prior to the actual saving of the object definition:

  - The object and characteristic names are valid names, i.e., they start with an alphabetic character (a-z).and do not contain an illegal characters.
  - The value of a characteristic must be one of the following:
  - Numeric: either real or integer (with values which can be contained in a long or double precision).
  - Alphabetic: the value must be in the alphabetic dictionary.

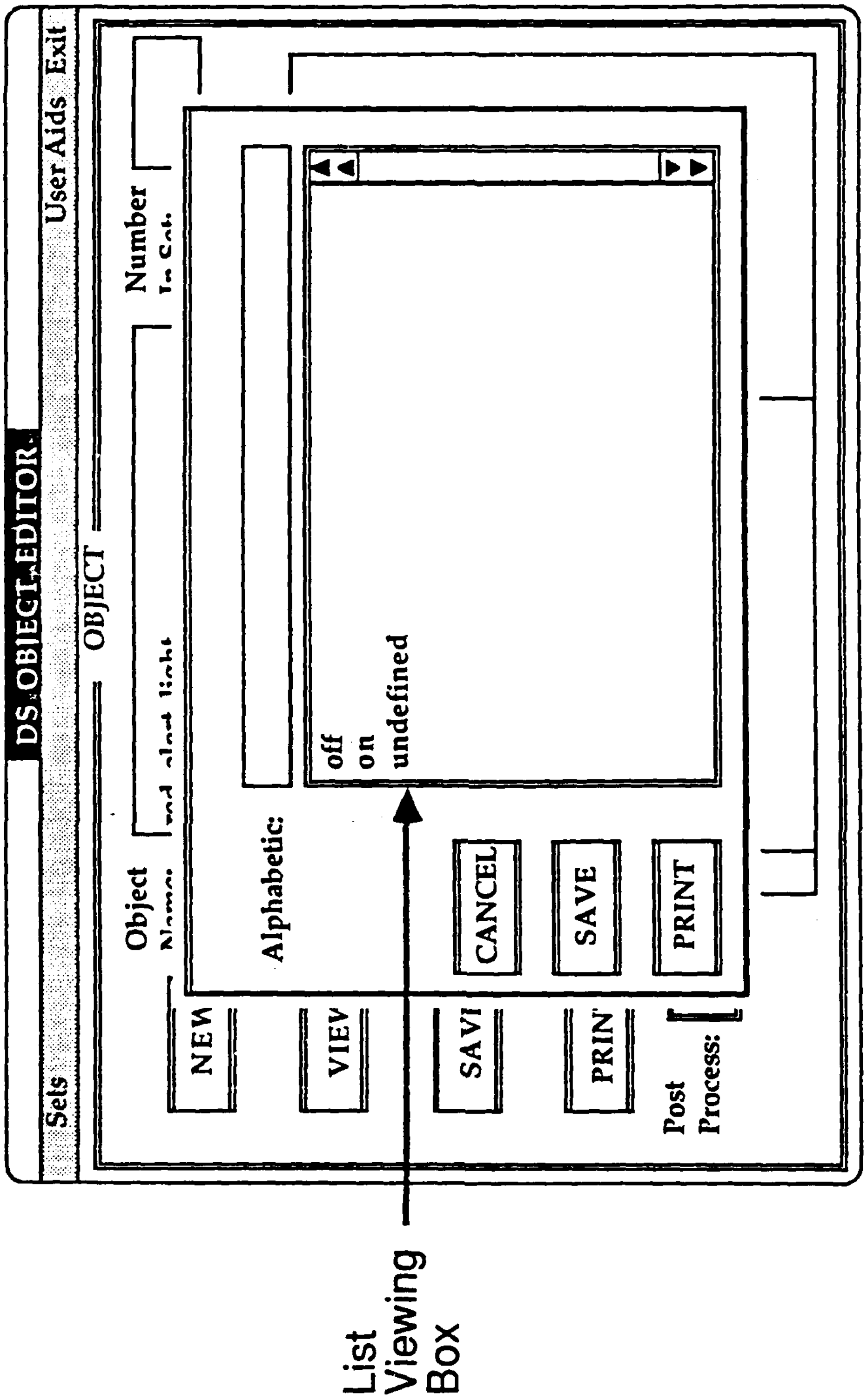


Figure 6-9 Edit Object Windows

- The characteristic type is determined based upon the entered value as described in Section 6.2.2.7 The first letter of the type (W=whole, D=decimal; A=alphabetic) is displayed in the type field of the appropriate characteristic. The default type is WHOLE.
- An initial value must be entered for every characteristic name.
- For every entered value, a characteristic name was supplied.
- If both the characteristic name and value fields are blank, that row of information is ignored.

- **PRINT** — prints the current object definition on the printer.

The object dialog window contains the following text entry boxes:

- **Object Name** — entry of the object name as a maximum of 28 characters.
- **Number In Set** — entry of the number of members in a set. Used only when the user selects the **Save a Set** option from the **Sets** menu bar. Valid entries are a number between 2 and 999.
- **Characteristic Name/Value** — entry of a maximum of 14 pairs of characteristic names and values. The type column is automatically filled in by EDIT_OBJ during the SAVE operation.

When the user selects the **Add an alphabetic** option from the **User Aids** menu bar, an alphabetic dialog window is displayed as shown in Figure 6-10. The alphabetic dialog window contains a list viewing box that displays the list of the currently defined alphabetic in alphabetic order. The user can select from the following pushbutton:

- **CANCEL** — cancels the add an alphabetic function.
- **SAVE** — adds the entered alphabetic name to the alphabetic dictionary.

The alphabetic dialog window also contains a text entry box for entry of the alphabetic name.

Input/Output

The files produced by EDIT_OBJ include the following and are illustrated below in Figure 6-10.

- Objects.o$ — Object library.
- Hosprop.p$ — Alphabetic dictionary.
- HOSSET.S$ — Set information file containing number of members and Snnnnnn.set (n is between 0 and 9999999) which will be created and used at simulation execution time to contain set information.

Each defined object requires 668 bytes in the objects.o$ file.

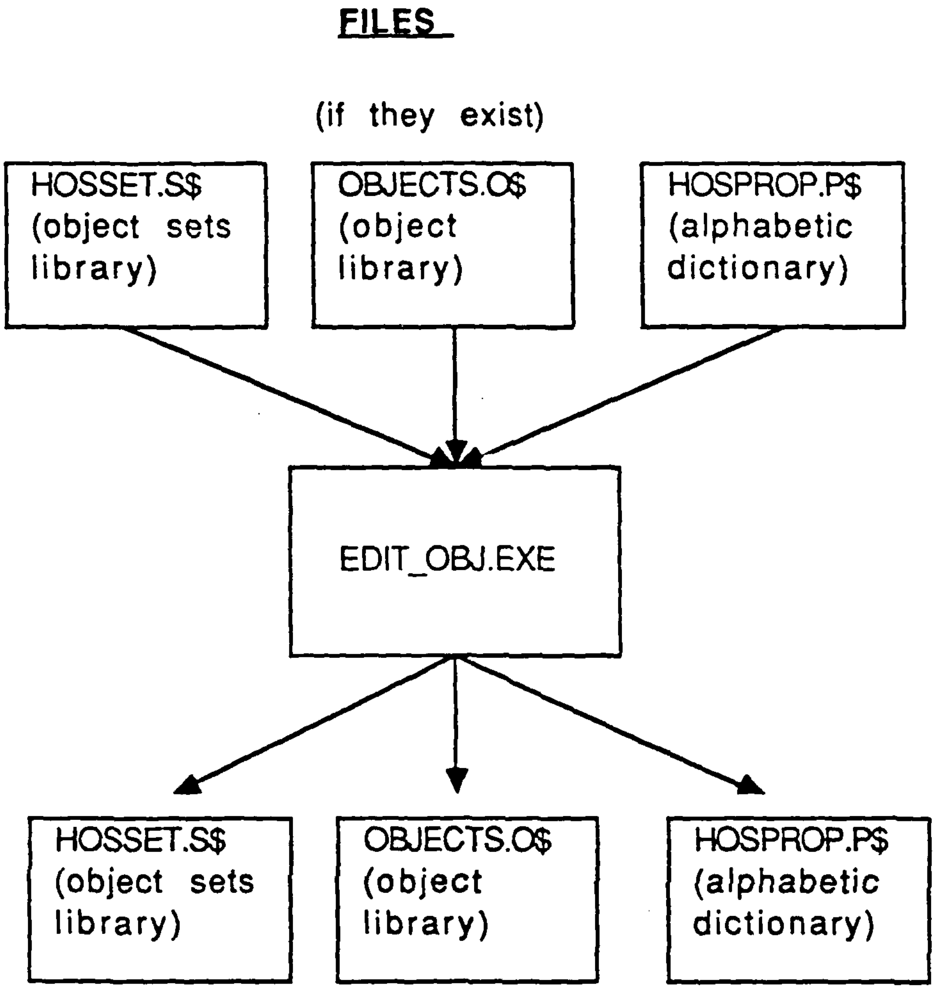

FILES
(if they exist)
HOSSET.S$ (object sets library)
OBJECTS.O$ (object library)
HOSPROP.P$ (alphabetic dictionary)
EDIT_OBJ.EXE
HOSSET.S$ (object sets library)
OBJECTS.O$ (object library)
HOSPROP.P$ (alphabetic dictionary)

Figure 6-10: Relation of Files Generated by EDIT_OBJ.

Error handling

OBJECT errors include the following :

- Failure to open a necessary file. This returns to the calling procedure, but does not cause program failure.
- File read/write failure. closes all files and returns but does not cause program failure.
- File seek failure. Since random file access is used wherever possible, seek faults can occur. This error closes all files and returns to the calling program, without doing a read, but does not cause program failure.
- All user errors are displayed to the user in a message or information window.
- Memory allocation errors which are recoverable.

6.2.2.8 HAL — HOS-IV Action Translator

HAL, the HOS-IV Action translator, translates actions into C code. It is invoked by the action editor, ACTEDIT, automatically when the user selects the **Translate** option on the **File** pull-down menu as described in Section 6.2.2.6.

Description

The HAL translator is a one pass translator using a forward-chaining translation scheme. The user ACTION file (Pnnnnnn.HPL) which is currently open within the action editor is input to HAL by EDIT_ACT. nnnnnnn represents a unique seven-digit number assigned by the action editor to translate the 28 character action name into a valid DOS file name. The translator reads one token at a time. The token is read in as a string which is isolated by valid delimiters such as the comma or a space. Each token is analyzed to determine its type -- for example, it determines if the current token is a HAL verb keyword, an object name, a characteristic name, or a local variable. Depending on the type of token, the translator will analyze the following token and determine if a statement is syntactically correct.

HAL Screens

HAL runs solely as a batch process with all input and output controlled by the action editor.

Input/Output

The HAL translator produces four files:

- The SYMBOL TABLE file which lists the name and type of the variables used by the action currently being translated. This file is accessible to the user from the VIEW FILE option of the User Aids pull-down menu and is named SYMBOL TABLE.
- The TRANSLATOR OUTPUT file which lists the HAL code and the results of the HAL translation including appropriate syntax error messages. This file is accessible to the user from the VIEW FILE option of the User Aids pull-down menu and is named TRANSLATOR OUTPUT.
- The C statement file which lists the translated HAL code (simname\pnnnnnnn.c) where nnnnnnn is the same seven-digit number assigned to the action name.
- The include file which contains local variable data definitions.

These files can be accessed through (1) the action editor by using the VIEW FILE option on the User Aids pull-down menu; (2) by opening the file directly through the file menu in the action editor; or (3) by typing it using DOS commands. The files are located in the \HOSIV\ HOSHPL subdirectory.

6.2.2.9 CREATE — Create Simulation

The CREATE module constructs the HOS-IV simulation based upon the entered events, rules, objects, and actions. It processed all the individual definitions to ensure that each referenced item has been defined. These checks include

- Ensuring that all actions referenced in events have been defined
- Ensuring that all actions, alphabetics, and object-characteristic pairs referenced in rules have been defined.
- Ensuring that all object-characteristic pairs, alphabetics, actions, and rules referenced in actions have been defined.

If all cross-references have been validated, then the C code generated for each referenced action is compiled and linked with the HOS-IV simulation C code and producing the executable simulation file. CREATE is illustrated in Figure 6-11.

6.2.2.10 RUNSIM — Run Simulation

RUNSIM executes the selected simulation and generates the simulation output files. It allows the user to indicate whether the simulation is to be started at the beginning of the simulation (i.e., the time specified via the SETUP module) or restarted from the point where the simulation was previously interrupted. RUNSIM creates a screen showing the simulation status at each time increment and the current active event, rule, and action. The RUNSIM screen contains the following pushbuttons:

- **PAUSE** — temporarily interrupt the simulation to freeze the current contents of the screen. As soon as the mouse is clicked again, the simulation resumes.
- **STOP** — suspend the current simulation at the displayed simulation time and save all simulation data so that the simulation can be restarted at a latter date.

6.2.2.11 RESULTS — View Results

The view results module is used to examine data produced by running a simulation. It is illustrated in Figure 6-12.

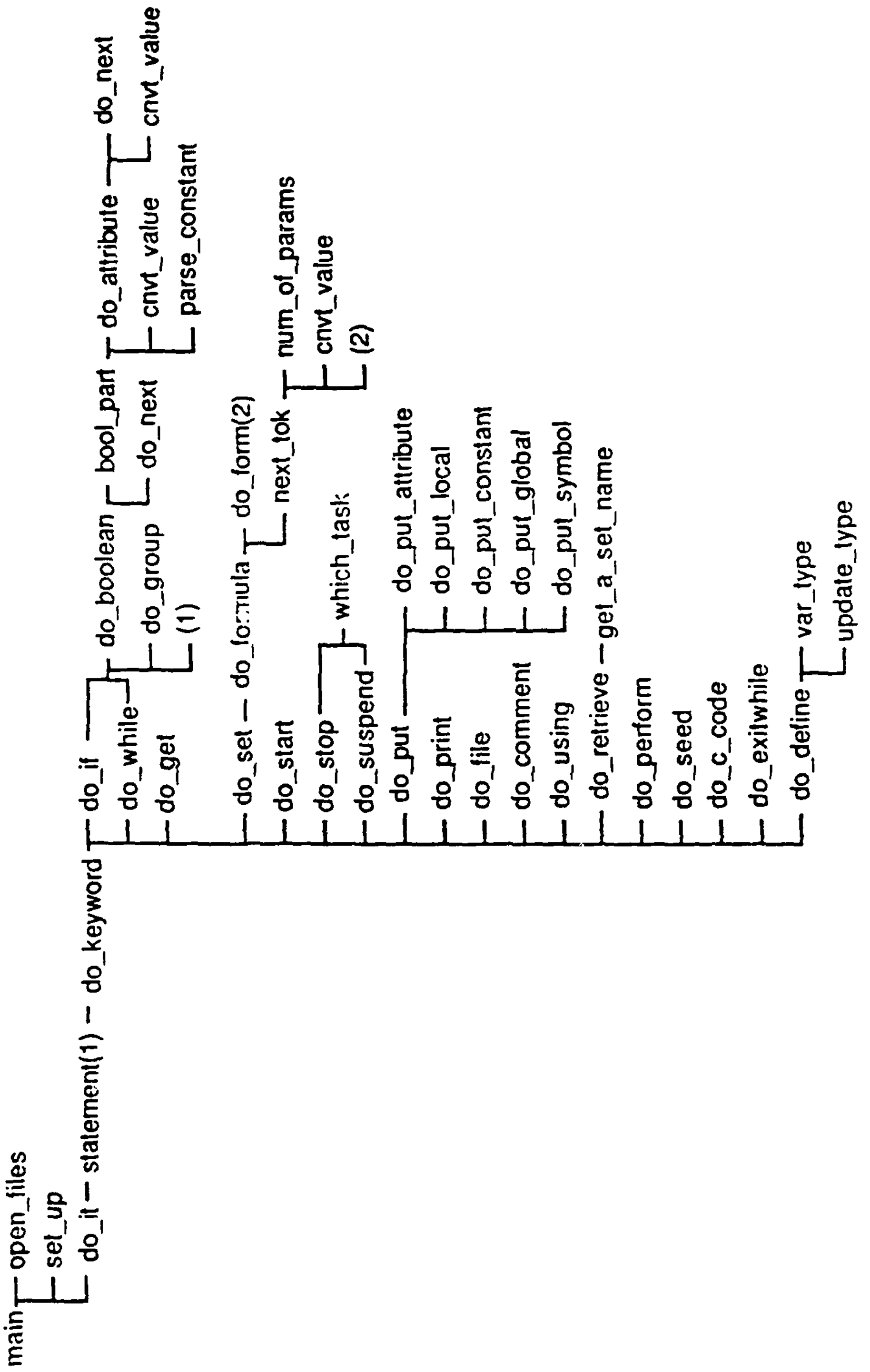


Figure 6-11: SIMLINK Functional Diagram.

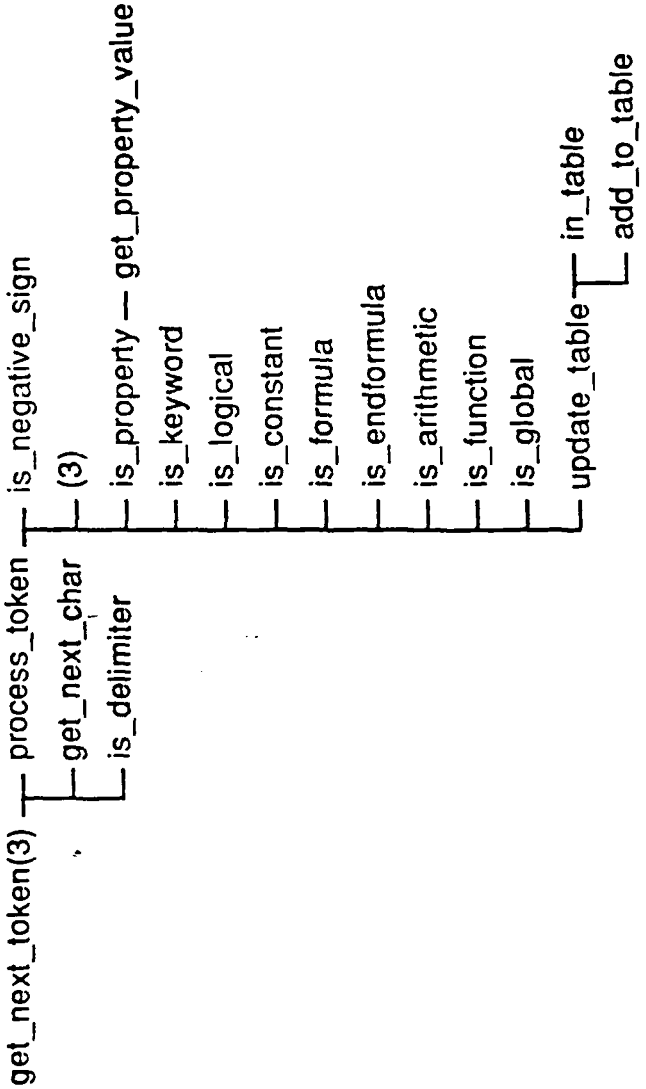


Figure 6-11: SIMLINK Functional Diagram - Cont.

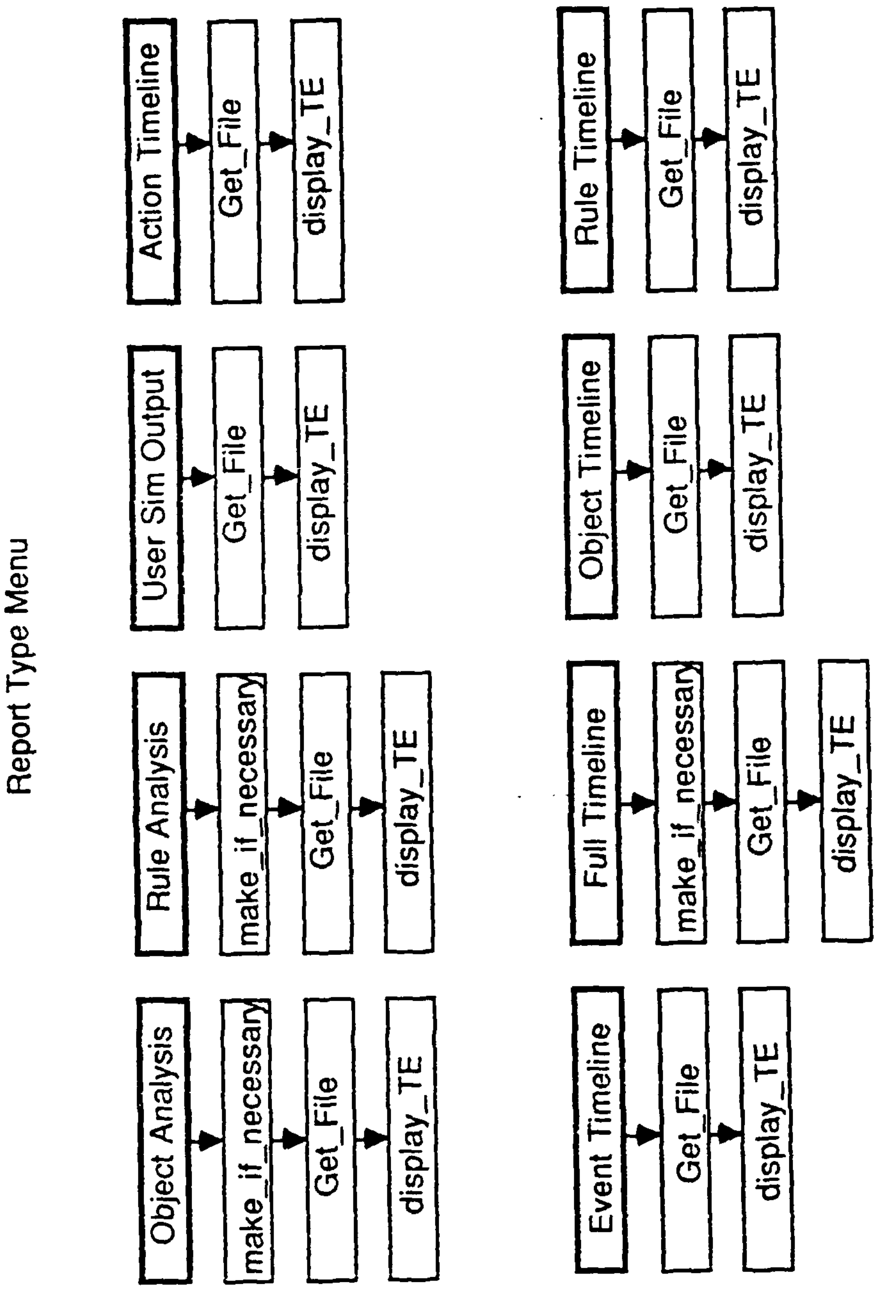


Figure 6-12: Schematic of View Results Module.

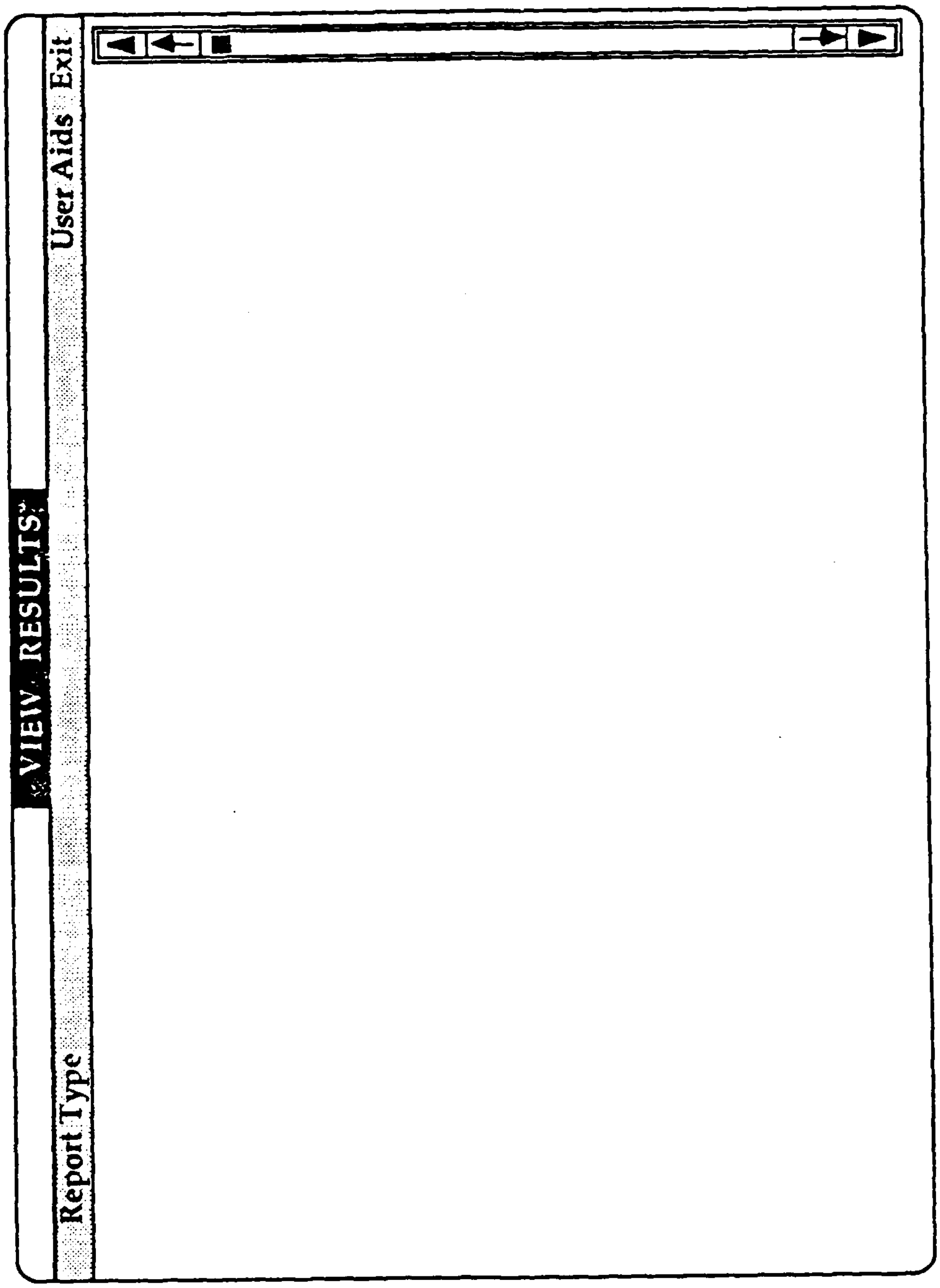


Figure 6-13: View Results.

Description

RESULTS contains a series of standard reports that are used to assist in the analyśis of simulation results. The available reports are the following:

- **Object Analysis** — contains the value of each object-characteristic pair at each time in the simulation when the value of the characteristic of the object changed.
- **Rule Analysis** — generates a statistics on rule usage including the number of time the rule was active and average duration of the activity.
- **User Simulation Output** — user-defined report produced by using the FILE verbs contained in actions.
- **Action Timeline** — generates a timeline showing the name of each active action at each time interval in the simulation.
- **Event Timeline** — generates a timeline showing the simulation time and event name of each active event.
- **Full Timeline** — combines the event, rule, action, and object timeline into one single report.
- **Object Timeline** — generates a timeline showing the simulation time when each characteristic of an object was modified.
- **Rule Timeline** — generates a timeline showing the simulation time and rule name of each active rule during the simulation.

RESULTS Screens

The view results module consists of a title bar, a menu bar, a text viewing window with a scrollbar, and a number of dialog boxes used for program interaction with the user as illustrated in Figure 6-13.

View Results Module Title Bar. The title bar contains the words 'VIEW RESULTS' as the function name in the center. It does not use the current activity or status information areas of the title bar.

View Results Module Menu Bar. The menu bar for the View results module contains the following menu options:

- **Report Type** — file related commands such as saving, opening, etc.
- **User Aids** — provides the capabilities to view help files and action files.
- **Exit** — terminates View results module and returns the user to the HOS-IV module.

The **Report Type** pull down menu contains:

- **Object Analysis** — opens the Object Analysis report; if it does not exist . the create missing report message window is displayed.
- **Rule Analysis** — opens the Rule Analysis report; if it does not exist the create missing report message window is displayed.
- **User Simulation Output** — opens the User Simulation Output report; if it does not exist the report missing message window is displayed.
- **Action Timeline** — opens the Action Timeline report; if it does not exist the report missing message window is displayed.
- **Event Timeline** — opens the Event Timeline report; if it does not exist the report missing message window is displayed.
- **Full Timeline** — opens the Full Timeline report; if it does not exist the create missing report message window is displayed.
- **Object Timeline** — opens the Object Timeline report; if it does not exist the report missing message window is displayed.
- **Rule Timeline** — opens the Rule Timeline report; if it does not exist the report missing message window is displayed.

The **User Aids** pull down menu contains commands that allow the user receive help on the current module and view action files.

- **Help** — allows the user to obtain additional information about using view results module.
- **View File** — allows the user to view action files created using the view results module.

View Results Module Text Viewing Window. The text viewing window consists of text area and a scrollbar. The text area is 23 rows by 77 columns. The scrollbar is to the right of the text area and has four control buttons and a relative file position indicator. The four controls on the scrollbar are:

- **Scroll Line Down** — displays a page of text starting from the line before the current top line.
- **Scroll Page Down** — displays a page of text starting one page before the current top line.
- **Scroll Page Up** — displays a page of text starting from the current bottom line.
- **Scroll Line Up** — displays a page of text starting from the line after the current top line.

View results module Dialog Boxes. If the selected report requires additional information, text entry boxes are generated to request the appropriate information.

View results module Message Windows. Message windows display information which the user must acknowledge by depressing one of the selected pushbutton. Some message windows may be cancelled.

- **File not found** — the indicated file contains the simulation results and could not be located.
- **Create missing report** — the user is requesting output of a report that has not yet been generated.
- **End viewing session** — confirmation that the view results functions is terminating.

View results module Information Windows. Information windows require no input from the user. They are informative messages indicating that the system is carrying out the entered command and to let the user know the status of the operation..

- **Reading file** — the file is being read

Input/Output

RESULTS uses the following files:

- **d:\htemp.txt** — temporary file created for saving and printing purposes
- **c:\hoslv\hoshelp\vrt_help.000** — list of help topics specific to the View Results module
- **c:\hoslv\hoshelp\vrt_help.nnn** — help files (numbered extension starting from 001)
- **c:\hoslv\currsim.dat** — name of the currently selected simulation.
- **c:\hoslv\***currsim\currsim***.dob** — simulation log for object activity.
- **c:\hoslv\***currsim\currsim***.dpr** — simulation log for action activity.
- **c:\hoslv\***currsim\currsim***.dev** — simulation log for event activity.
- **c:\hoslv\***currsim\currsim***.dtk** — simulation log for task activity.
- **c:\hoslv\***currsim\currsim***.tlm** — simulation timeline.
- **c:\hoslv\***currsim\currsim***.log** — user-defined simulation output generated from FILE verbs.
- **c:\hoslv\***currsim\currsim***.rul** — simulation rule information.
- **c:\hoslv\***currsim\currsim***.oba** — simulation object information.

7. STATIC SIMULATION - CAR

The Crewstation Assessment of Reach (CAR) is an anthropometric model which predicts a population's or an individual's ability to be properly positioned in a user-defined workspace and evaluates the reachability of all controls (Harris et al, 1980; Bennett et al., 1982; Harris and lavecchia, 1984). CAR was initially developed by the U.S. Navy in 1976 as a software tool for predicting population accommodation for proposed cockpit designs. CAR is intended to be used in the early design phase of a system's development before mock-ups are constructed and before any undesirable design features become fixed. CAR has typically been applied to aviator populations but has been applied to maintainer populations as well.

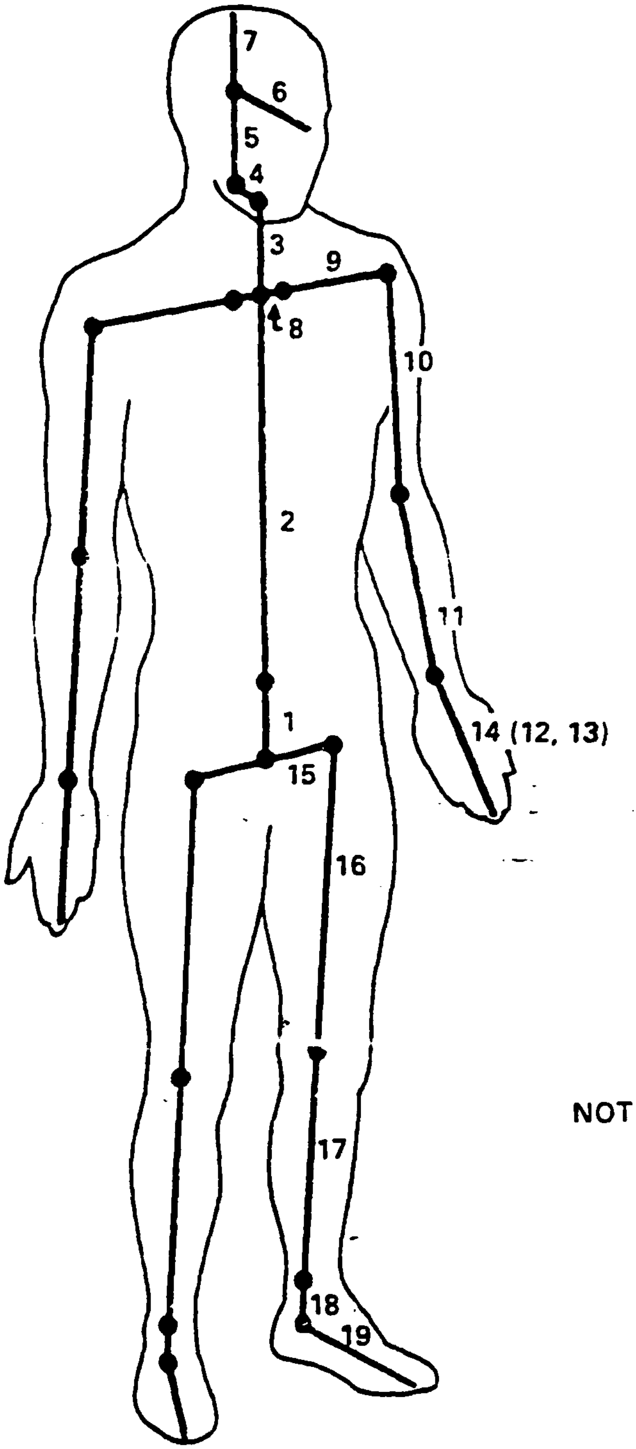
CAR transforms operator body measurements into internal links which roughly correspond to major skeletal bones as illustrated in Figure 7-1. When simulating the reach for a control, CAR positions the operator at the user-defined anchorage point in the workspace and adds successive skeletal links in the direction of the control, applying appropriate angular limits of motion at each link joint. CAR predicts the percentage of the population that can reach each control and the average distance by which unreachable controls are missed. CAR can perform this accommodation analysis for specific individuals or for a sample of soldiers generated from population statistics using a Monte Carlo process.

CAR consists of the following three main modules:

- Operator definition;
- Workstation configuration definition; and
- Accommodation analysis.

The structure of the CAR program and data flow is illustrated in Figure 7-2. The fourth generation of CAR, known as CAR-IV, is the latest implementation. The CAR-IV software is available in versions for the CDC 6600/Cyber 176, DEC VAX, and IBM PC.

The CAR modules will be restructured within MMP6 with the soldier definition portion becoming part of HIG (Section 5) and the remainder becoming part of STATSIM. The techniques used in CAR to generate an operator population from statistical data about the target population can be used not only to generate body measurements to evaluate operator accommodation within the workspace but also to generate performance measurements to evaluate operator performance. The CAR routines that are used to generate the operator samples will be extracted



1 – LUMBAR
2 – THORACIC
3 – NECK (VERTICAL)
4 – NECK (HORIZONTAL)
5 – HEAD (LOWER)
6 – EYE
7 – HEAD (UPPER)
8 – INTERCLAVICULAR
9 – CLAVICULAR
10 – HUMERAL
11 – RADIAL
12 – HAND (CLENCHED)
13 – HAND (FINGERTIP GRIP)
14 – HAND (EXTENDED)
15 – PELVIC
16 – FEMORAL
17 – TIBIAL
18 – ANKLE
19 – FOOT

NOTE: LINKS 8–19 HAVE BOTH
RIGHT AND LEFT LINKS

Figure 7-1: Illustration of Major Skeletal Measurements.

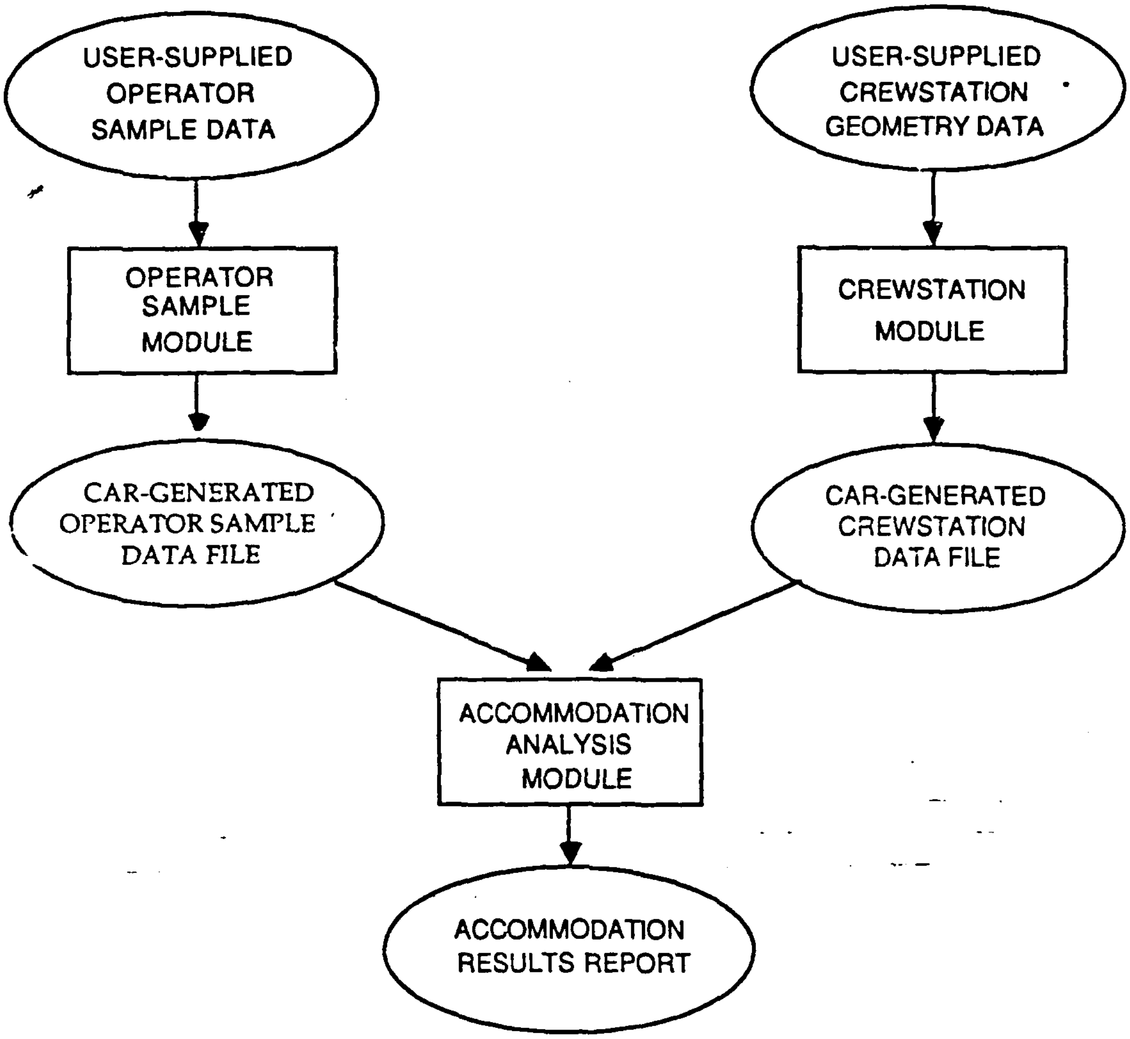


Figure 7-2: Structure of Static Simulation and Data Flow.

and modified to generate either body or performance measurements for the desired soldier sample. The workspace geometry and reach accommodation analysis portions of CAR will be integrated into STATSIM.

The techniques and algorithms included in CAR have undergone extensive validations. A preliminary evaluation of CAR hand reach capabilities was performed in 1982 in order to determine the accuracy of CAR's predictions for male populations. This was accomplished by comparing CAR-generated hand reach envelopes with experimentally derived anthropometric reach envelopes that were collected by Kennedy at Wright-Patterson Air Force Base for 5th, 50th, and 95th percentiles (Kennedy, 1978). The CAR-generated reach envelopes most closely matched the Kennedy envelopes in the regions where most critical aircraft controls are located — that is, to the forward and side cockpit regions (Bennett et al., 1982).

The Monte Carlo technique for generating a sample of operators from population statistics was also validated as part of the 1982 validation effort (Bennett et al., 1982). The means and standard deviations of the Kennedy subject pool were entered as population data to CAR with the intercorrelation data supplied by the 1967 Air Force male data. The anthropometric characteristics of the CAR-generated sample were very similar to those of the subjects used by Kennedy as indicated by similar means and standard deviations of body measures for the Kennedy and CAR groups.

A hand reach envelope validation study was also performed for female populations comparing CAR-generated results with Kennedy's reach envelopes (Iavecchia & Harris, 1984). Initially, CAR utilized male link transformation equations to predict female body dimensions of females. This study revealed significant discrepancies between the CAR and experimental reach envelopes. As a result, new transformation equations for female operators were developed based upon cadaver studies (Trotter & Gleser, 1952). The reach validity tests were rerun with significant improvements obtained for 5th and 50th percentile females, while lesser improvement was achieved for the 95th percentile females.

A validation study was conducted during 1985 and 1986 with the purpose of evaluating CAR's ability to predict accommodation for an actual set of operators representative of Navy aviators in an actual Navy crewstation. CAR's predictions of the vision and reach accommodation of 10 male subjects and 2 female subjects in an F-14 cockpit were compared to data collected on the reach of each subject in the F-14 cockpit. A set of 37 controls was selected for the study, which required the use of the three types of grips modeled by CAR. The F-14 cockpit selected for the experiment was equipped with a standard seat and harness restraint system. Experimental data were collected for the effect of flight clothing and survival vests, hand and foot control reaches, various harness constraint conditions, vision positioning, and seat adjustment. The initial validation study included analysis of male hand reaches. The results of the validation study indicate that CAR is an excellent predictor of population accommodation and is able to closely predict experimental reach outcome, average miss distance for reach failures, and the direction of and average horizontal and vertical displacement from the DEP. Generally, where CAR results differed from experimental results, CAR tended to err conservatively (i.e., CAR predicted reach failure where success had occurred or CAR overpredicted the reach miss distance).

7.1 Static Simulation

The Static Simulation (STATSIM) will allow the user to define the geometry of a workspace and run an accommodation analysis to determine if a selected set of hypothetical individuals can be accommodaied in the crewstation.

7.1.1 Crewstation Definition

Crewstation geometry data which the user must input to STATSIM includes crewstation anchorage point, control locations, control grip type, and head clearance data. The user can enter data in his/her own coordinate system with an option for inches or centimeters as the measurement units. STATSIM will transform all data into a geometric shell of the workspace. For each simulation, the initial body posture (i.e., sitting or standing) and an anchorage point must be specified. The anchorage point serves as a fixed location in space where the STATSIM model anchors a specific body part of the link-person. For seated operators, anchorage options include:

> A fixed seat that places the base of the operator's spine at the Seat Reference Point (SRP);
>
> A variable seat that places the operator's eye at the Design Eye Point (DEP) and along a line-of-sight (LOS) angle based upon the range of seat adjustment; and
>
> A foot control that places the heel of the operator's right or left leg at the location of that control.

For standing operators, anchorage options include foot control and hip or shoulder tethering.

The user must provide detailed information relevant to the crewstation anchorage type such as a Seat Reference Point (SRP), seat back and pan angles, and seat travel in one or two axes for seated anchorages. The Seat Reference Point is the center of the line segment formed by the intersection of the seat back and the seat pan. The range of seat travel is defined with respect to the SRP by entering the coordinates of the furthest down-forward position of the seat and the furthest upward-back position of the seat. For standing anchorages, the user enters the location of the foot, shoulder, or hip anchor point. A Design Eye Point (DEP), line-of-sight angle (LOS), and head clearance data are also entered. The user also enters a harness percentage which specifies where the harness falls along the operator's shoulder.

For each control, the user enters the crewstation location (e.g., directly from a blueprint) as well as the body part (right/left hand or foot) which will manipulate the control, the required grip posture for hand controls, and the harness condition (locked or unlocked). STATSIM models three types of grip postures -- **clenched** (for a control stick or throttle); **fingertip** (for a knob or locked toggle); and **extended fingertip** (for a pushbutton or toggle). An additional point that represents the limit of linear range of movement is specified for adjustable controls such as throttles.

7.1.2 Accommodation Analysis

STATSIM analyzes the ability of an operator to reach a control by positioning the appropriate operator body part at the anchorage point and then adding successive links toward the control. The links are constrained by angular limits of motion assigned to each link joint, the harness conditions and the type of clothing. Three clothing options are provided — "shirt-sleeve"; summer flight clothing; and winter flight clothing. The clothing option selected will affect both link lengths and angular limits of motion. During the accommodation analysis, STATSIM determines the vision and reach accommodation of every operator in the sample. Depending on the crewstation anchorage, STATSIM employs different procedures for anchoring an operator in the workspace before simulating reach attempts. For example, for a seated anchorage with no seat travel, the base of the lumbar joint will be placed a particular distance above the seat pan and in front of the seat back based upon calculated body enfleshment for the operator. STATSIM then adds links successively in the direction of each control, applying angular limits of motion at each joint.

For every control, three types of reach are simulated — **Zone 1** where the shoulder is fixed against the seat back (such as would occur in a high-G environment); **Zone 2** where the operator strains against a harness in reaching a control; and **Zone 3** where the harness is unlocked. For each zone of reach, STATSIM outputs:

- The percentage of the population who could reach each control,
- The average and worst case distance by which each control was missed, and
- A correction vector for the average and worst case misses which can be added to the original control location to move the control closer to the operator and thereby improve population accommodation.

The latter piece of information can be used directly by the crewstation designer reconfiguring the workspace. STATSIM can be rerun in an iterative fashion to gain a new assessment of the population accommodation for the corrected control position. This iterative process can continue until the desired population accommodation is achieved.

STATSIM also computes the following vision accommodation data:

- The percentage of the population whose monocular eye (i.e., midpoint between eyes) can be positioned at the DEP;
- The average horizontal and vertical displacement of the operators' monocular eye from the DEP;
- The percentage of the population whose monocular eye fell forward of the DEP;
- The percentage of the population whose monocular eye fell above, on, or below the LOS; and
- The average horizontal and vertical seat travel necessary to move the operator's monocular eye to the DEP and/or along the LOS for seated anchorages with adjustable seats.

7.1.3 STATSIM

STATSIM is organized into the following modules:

- **SIMMAIN**: high-level system controller that manages the lower-level functions and interacts with the user to invoke the lower-level functions.
- **DEFWORK**: maintains the tables associated with the workspace configuration including creation of new workspaces, modification of existing workspaces, and deletion of existing workspaces.
- **ANALYZE**: executes an accommodation analysis for the selected sample and the defined workspace.
- **RESULTS**: generates accommodation analysis results for printing or screen viewing.

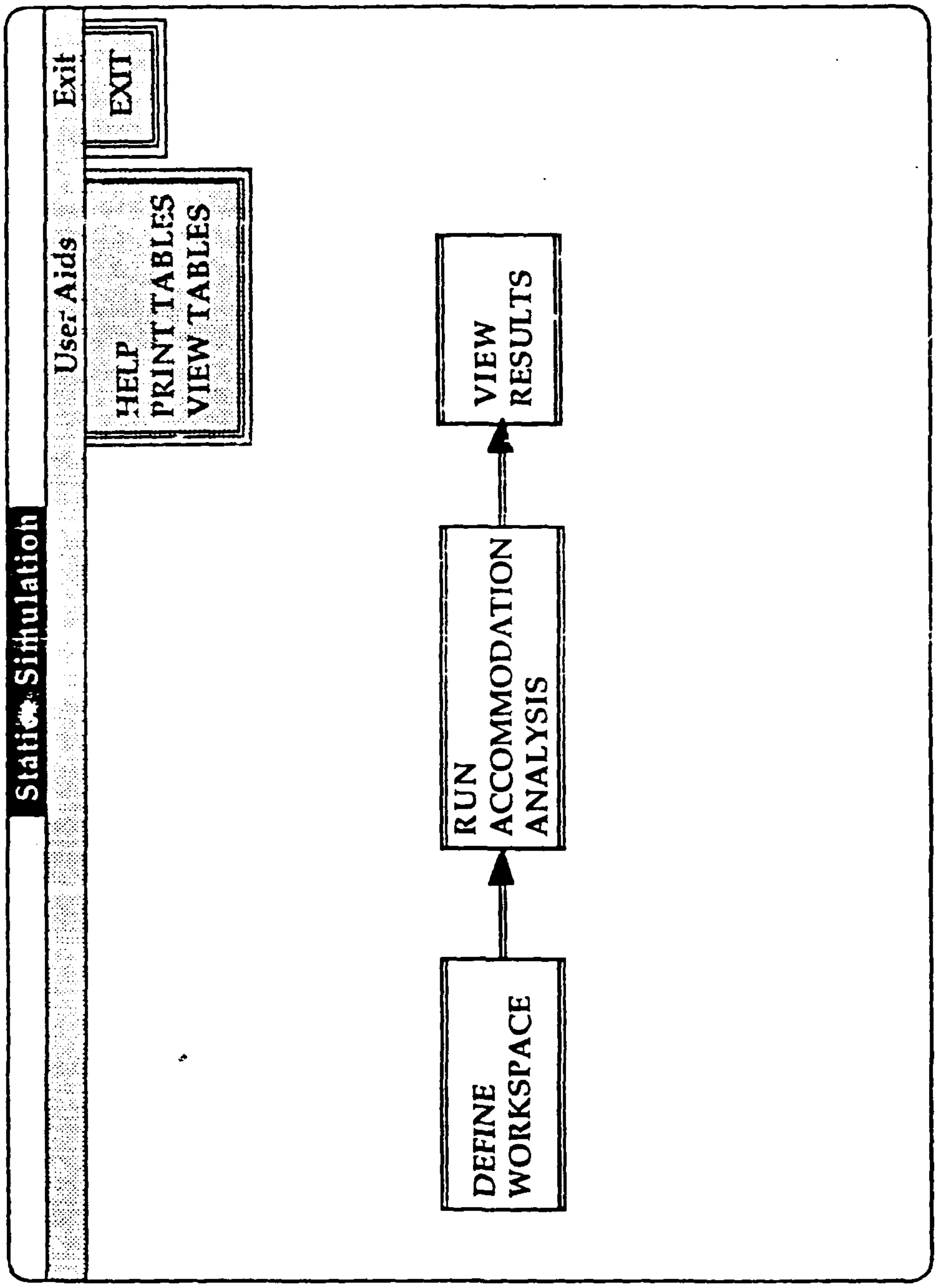


Figure 7-3: Static Simulation Main Screen

The organization of the static simulation is presented in Figure 7-3 with the left to right progression showing the sequence in which the modules must be invoked in order to identify properly the components necessary to run an accommodation analysis.

7.1.3.1 SIMMAIN

SIMMAIN is the main module in STATSIM and is invoked when STATSIM is invoked. It is the top level module that controls interactions with all other modules of the STATSIM system. There are four commands which can be executed from the main screen of STATSIM — Define Workspace, Define Specify HIG Sample, Run Accommodation Analysis, and View Results. A pushbutton was created for each function and the user invokes the desired function by moving the mouse over the title of the selected pushbutton and clicking the mouse. Whenever the user terminates one of the lower level modules, control is returned to SIMMAIN for the selection of the next module or to return to the top-level SEPTA system screen.

SIMMAIN Screens

The SIMMAIN screen contains a diagram of pushbuttons and arrows indicating the functional flow of steps invoked in defining and generating a sample as illustrated in Figure 7-3.

SIMMAIN Title Bar. The title bar of SIMMAIN contains the words Static Simulation as the function name. SIMMAIN does not use the current activity or status information areas of the title bar.

SIMMAIN Menu Bar. SIMMAIN contains the following menu options of the menu bar as illustrated in Figure 7-3:

- **User Aids** — provides the capabilities to view files and obtain help.
- **Exit** — terminates SIMMAIN and returns the user to the MMP6 main menu.

The **User Aids** pull-down menu contains the following commands:

- **Help** command that allow the user to obtain help windows about the SIMMAIN module
- **Print Tables** command that allows the user to obtain a copy of any of the STATSIM tables on the printer
- **View Tables** command that allows the user to obtain a copy of any of the STATSIM tables on the screen.

SIMMAIN Main Screen. The SIMMAIN main screen contains the following pushbuttons:

- **Define Workspace**— calls the DEFWORK module to allow the user to define the necessary parameters to define the workspace configuration.
- **Run Accommodation Analysis** — calls the ANALYZE module to evaluate the ability of the selected population to be accommodated in the workspace.
- **View Results** — calls the RESULTS module to obtain the results of the accommodation analysis.

7.1.3.2 DEFWORK

Description

The DEFWORK module of STATSIM maintains information for all user-defined workspaces. This module is used to specify the user's coordinate system and the workspace geometry. The workspace definition consists of an anchorage point, vision parameters, an operator seat, an overhead canopy, and a set of controls which the operator is expected to reach.

DEFWORK Tables

DEFWORK maintains the database tables containing the workspace description. The elements of the database tables described below contain the R:BASE V field types as text, real, note, or integer. The Workspace table (WRKSPACE) contains the following information:

- Workspace Name: A text name containing a maximum of 28 characters. This name is used to link all tables about a particular workspace together and appears in all list selection boxes that list the defined workspaces.
- Workspace Description: A text name containing a maximum of 80 characters.
- Workspace Comments: A note field containing any general comments about the particular workspace.
- Units Type: A 12 character text description containing either the word 'inches' or 'centimeters' to indicate the units of measurements for all data related to the workspace.
- Anchorage: A 20 character text description containing either one of the following to indicate the anchorage type: design eye point, seat, foot-sitting, foot-standing.
- Foot Anchorage-X: A real number containing the x-coordinate of the foot control point if foot anchored.
- Foot Anchorage-Y: A real number containing the y-coordinate of the foot control point if foot anchored.
- Foot Anchorage-Z: A real number containing the z-coordinate of the foot control point if foot anchored.
- Which Foot: A 5 character text description indicating which foot is anchored — right or left.

- Design Eye Point-X: A real number containing the x-coordinate of the design eye point.
- Design Eye Point-Y: A real number containing the y-coordinate of the design eye point.
- Design Eye Point-Z: A real number containing the z-coordinate of the design eye point.
- Design Eye Point Above-X: A real number containing the x-coordinate of a point directly above the design eye point.
- Design Eye Point Above-Y: A real number containing the y-coordinate of a point directly above the design eye point.
- Design Eye Point Above-Z: A real number containing the z-coordinate of a point directly above the design eye point.
- Design Eye Point Forward-X: A real number containing the x-cooidinate of a point directly in front of the design eye point.
- Design Eye Point Forward-Y: A real number containing the y-coordinate of a point directly in front of the design eye point.
- Design Eye Point Forward-Z: A real number containing the z-coordinate of a point directly in front of the design eye point.
- Line of Sight Angle: A real number containing the line-of-sight angle.
- Seat Back Angle: A real number containing the seat back angle (degrees back from vertical).
- Seat Pan Angle: A real number containing the seat pan angle (degrees up from horizontal).
- Harness Location: A real number containing the location of any harness along the shoulder expressed as a percentage of the distance from midline to shoulder.
- Seat Track: A 40 character text description indicating the seat track as one of the following: no track, single track: down-back to down-forward, down-back to up-back, double track.
- Seat Reference Point Up Back-X: A real number containing the x-coordinate of the seat reference up back point.
- Seat Reference Point Up Back-Y: A real number containing the y-coordinate of the seat reference up back point.
- Seat Reference Point Up Back-Z: A real number containing the z-coordinate of the seat reference up back point.
- Seat Reference Point Down Forward-X: A real number containing the x-coordinate of the seat reference down forward point.
- Seat Reference Point Down Forward-Y: A real number containing the y-coordinate of the seat reference down forward point.
- Seat Reference Point Down Forward-Z: A real number containing the y-coordinate of the seat reference down forward point.
- Seat Reference Point Down Back-X: A real number containing the x-coordinate of the seat reference down back point.

- Seat Reference Point Down Back-Y: A real number containing the y-coordinate of the seat reference down back point.
- Seat Reference Point Down Back-Z: A real number containing the z-coordinate of the seat reference down back point.
- Minimum Head Clearance: A real number containing the minimum head clearance.
- Helmet Thickness: A real number containing the helmet thickness.
- Head Clearance Point-X: A real number containing the x-coordinate of the head clearance point on the canopy.
- Head Clearance Point-Y: A real number containing the y-coordinate of the head clearance point on the canopy.
- Head Clearance Point-Z: A real number containing the z-coordinate of the head clearance point on the canopy.

The Workspace Control table (WCONTROL) contains the following information:

- Workspace Name: A text name containing a maximum of 28 characters. This name is used to link all tables about a particular workspace together and appears in all list selection boxes that list the defined workspaces.
- Number of Controls: An integer number indicating the number of controls included in the workspace with a maximum of 50.

The following elements will be repeated for each control:

- Control Name: An 20 character text description that names the control.
- Control Description: An 80 character text description that describes the control.
- Control Location-X: A real number containing the x-coordinate of the control location.
- Control Location-Y: A real number containing the y-coordinate of the control location.
- Control Location-Z: A real number containing the z-coordinate of the control location.
- Body Part: A 10 character text description containing one of the following to indicate the body part used to manipulate the control: left hand, right hand, both hands, left foot, right foot, both feet.
- Grip: A 10 character text description containing one of the following to indicate the grip to be used to manipulate the hand controls: clenched, fingertip, extended, or blank if not applicable.
- Harness Condition: A 10 character text description containing one of the following to indicate the hardness condition used when manipulating hand controls: clenched, locked, unlocked, or blank if not applicable.
- Critical Control Indicator: A 10 character text description containing one of the following to indicate the if the controls is deemed critical: critical or noncritical.
- Adjustable Location-X: A real number containing the x-coordinate of the control's adjustable location.

- Adjustable Location-Y: A real number containing the y-coordinate of the control's adjustable location.
- Adjustable Location-Z: A real number containing the z-coordinate of the contrcl's adjustable location.

DEFWORK Screens

The DEFWORK screen initially contains the title and menu bars. As the user selects various menu options, the window portion of the DEFWORK screen will contain the appropriate dialog windows for the user to enter the indicated information.

DEFWORK Title Bar. The title bar of DEFWORK contains the words 'Static Simulation' as the function name. The words 'Define Workspace' will be inserted in the current activity area of the title bar. The status information areas of the title bar will contain various information depending upon the current activity.

DEFWORK Menu Bar. DEFWORK contains the following menu options of the menu bar as illustrated in Figure 7-4:

- **Workspace Options** — table related commands such as new table, edit table, save table, delete table, etc.
- **Data Type** — indicates the type of data to be entered or viewed as either workspace description, design eye point, seat, control, or head clearance data.
- **User Aids** — provides the capabilities to print sample information or obtain help messages.
- **Exit** — terminates DEFWORK and returns the user to the HIGMAIN main menu.

The **Workspace Options** pull-down menu contains commands that allow the user to create, save, and define new workspaces. It contains the following commands:

- **New Workspace** — defines a new workspace. It closes the current workspace after asking if the changes should be saved and then creates a new empty workspace. It clears the current dialogue window and puts the description dialogue window on the screen with the text cursor placed in the first character in the workspace name field.
- **Open Workspace** — opens the tables associated with an existing workspace. It closes the current workspace after asking if changes should be saved and then opens the workspace file selection dialog box. The user views the currently defined workspace names and then selects one. The screen is cleared and the workspace description dialogue window is placed on the screen with the contents of the description table placed in the appropriate fields. The text cursor is placed in the first character in the workspace name field.

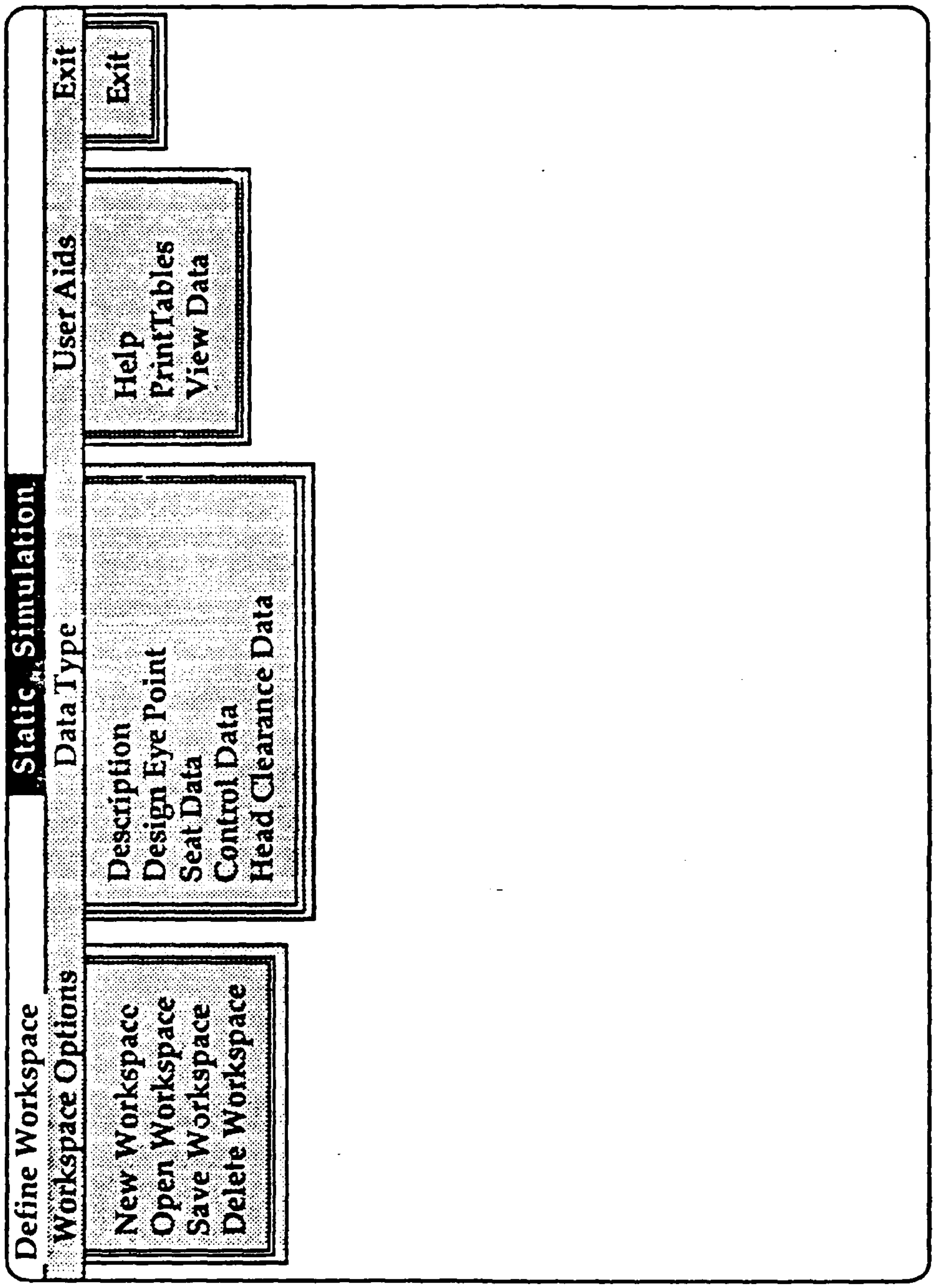


Figure 7-4: Static Simulation Screen Showing Windows.

- **Save Workspace** — saves the currently defined workspace. First, all validity and consistency checks are performed on the workspace definition. This checks include:
  - The down-forward point of the seat must be in front of the down-back point.
  - The up-back point of the seat must be above the down-back point.
  - The down-back — up-back track must be parallel to the vertical axis defined by the user.
  - The up-back point must be above the line between the down-forward and down-back seat points.
  - The DEP point must be above and in front of the seat.
  - The foot-anchoring ponit must be below and in front of the seat.

  If any errors are detected, message windows will be displayed indicating the nature of the error and suggested corrective actions. Once the workspace data is judged error-free, then the save dialog window will be displayed for the user to specify the name of the workspace file. All the tables associated with the workspace will be saved. An informative message window is displayed showing the workspace tables being created.
- **Delete Workspace** — deletes the currently selected workspace. A confirmation message window is displayed to request user confirmation before the delete is performed.

The **Data Type** pull-down menu contains commands that allow the user to specify the type of information to be currently displayed on the screen for input or editing. It contains the following commands:

- **Description** — generates the description dialogue window.
- **Design Eye Point** — generates the design eye point dialogue window.
- **Seat Data** — generates the seat data dialogue window.
- **Control Data** — generates the control data dialogue window.
- **Head Clearance Data** — generates the head clearance data dialogue window.

The **User Aids** pull-down menu contains the following commands:

- **Help** command that allow the user to obtain help windows about the DEFWORK module.
- **Print Tables** command that allows the user to obtain a copy of any of the workspace tables on the printer.
- **View Data** command that allows the user to obtain a copy of any of the workspace workspace tables on the screen.

DEFWORK Windows.

The DEFWORK workspace description window is used for entering the necessary information to describe the basic components of the workspace including the name, description measurement units, anchorage, and number of controls. The anchorage defines a fixed point in space to which the operator is to be positioned. Once positioned for anchorage, STATSIM fixes the position of the lumbar link and operator remains at the point for DEP, head clearance, and control reach analysis. STATSIM permits the following to be defined as anchorages:

- **Design Eye Point** anchorage will position a seated operator for vision. STATSIM will first attempt to position the operator's eye at the DEP. If unsuccessful, STATSIM will next attempt to place the operator's eye along the line-of-sight. If unable to position the operator on the LOS, STATSIM will attempt to place the operator above the LOS.
- **Seat anchorage** indicates that the operator will be placed in a fixed seat and no positioning adjustment occurs.
- **Foot-Seated** anchorage indicates that the heel of the operator's right or left foot will be positioned at the Foot Anchorage Point (FAP) at an angle within the angular limits of motion for the tibial link.
- **Foot-Standing** anchorage indicates that the operator's right or left foot will be positioned at the Foot Anchorage Point (FAP) and no positioning adjustment occurs.

The current activity area of the title bar will contain the words 'Define Workspace'. The workspace description window, as illustrated in Figure 7-5, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the workspace name field.
- **Save** — saves the current workspace description as displayed on the screen. The following validation is performed prior to the actual saving the workspace description:

  The workspace name and description fields are valid names, i.e., they start with an alphabetic character (a-z) and do not contain any illegal characters.

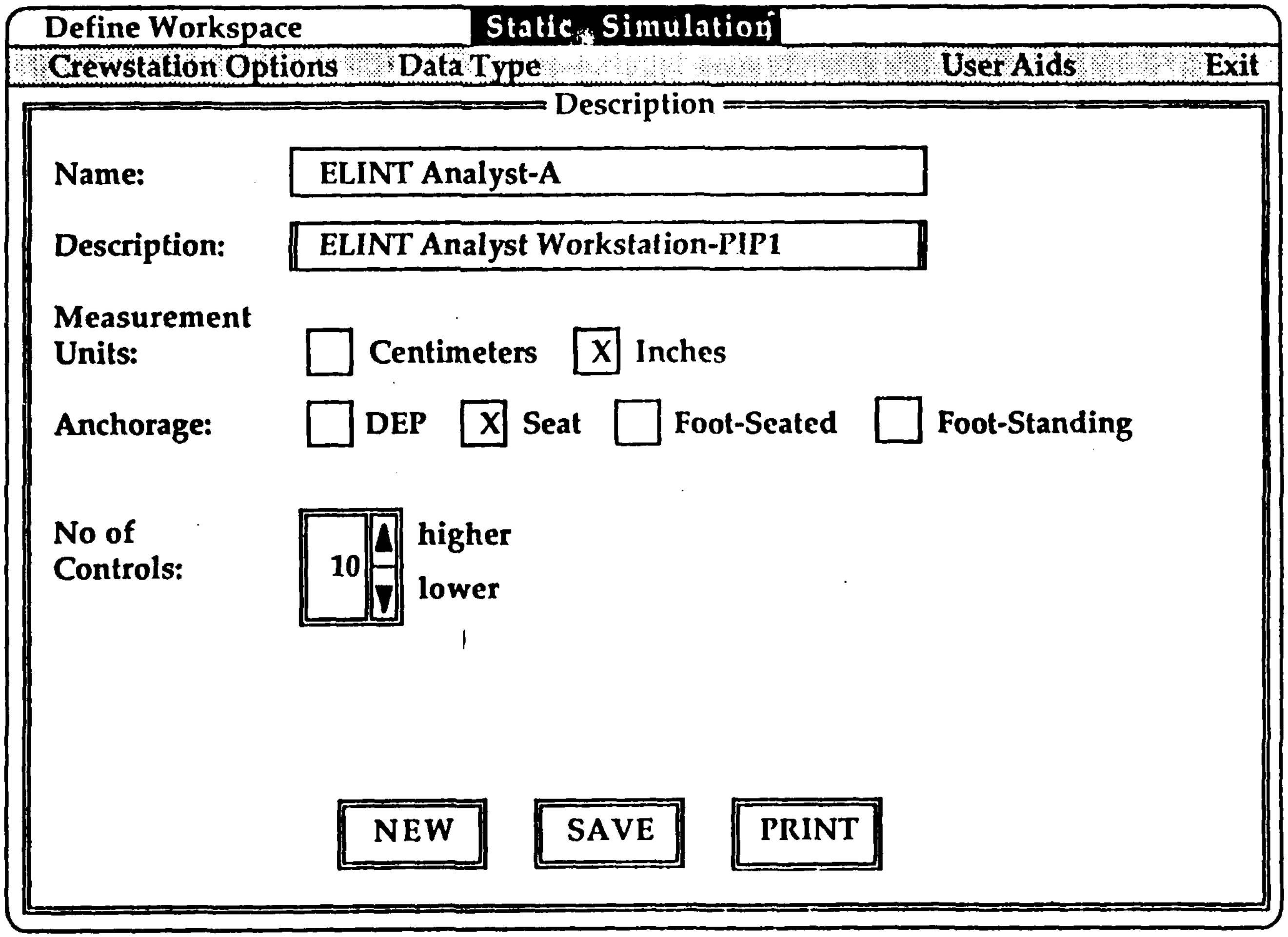
  The number of controls is an integer number between 1 and 50.
- **Print** — print the current workspace description on the printer.

The workspace description dialogue window contains the following text entry boxes:

- **Workspace Name** — entry of the workspace name as a maximum of 28 characters.
- **Workspace Description** — entry of the workspace description as a maximum of 80 characters.

The following check boxes are used in the workspace description dialogue window:

- **Measurement Units** — specify the units of measurements as either 'inches' or 'centimeters'.
- **Anchorage** — specified the anchorage as Design Eye Point, Seat, Foot-Seated, or Foot-Standing.



Name: ELINT Analyst-A

Description: ELINT Analyst Workstation-PIP1

Measurement Units: □ Centimeters ☒ Inches

Anchorage: □ DEP ☒ Seat □ Foot-Seated □ Foot-Standing

No of Controls: 10 ▲ higher ▼ lower

Figure 7-5: Workspace Definition.

The following click boxes are used in the workspace description dialogue window:

- **Number of Controls** — entry of the number of measurements included in the workspace as an integer number between 1 and 50.

The DEFWORK design eye point window is used for entering the vision information for a workspace. The current activity area of the title bar will contain the words 'Define Workspace'. The status information area contains the word 'Inches' to inform the user of the selected measurement units to be used for all input data. The STATSIM vision parameters are the coordinates of the design eye point and a line-of-sight angle. The LOS is a line in the DEP plane and is used as a criterion in adjusting the seat for optimal operator vision. For DEP-anchored workspaces, the seat is adjusted to place the operator at the DEP. If the seat adjustment is not sufficient to position the operator at the DEP or the workspace is not DEP-anchored, then STATSIM attempts to place the operator along the LOS.

The design eye point window, as illustrated in Figure 7-6, contains the following pushbuttons:

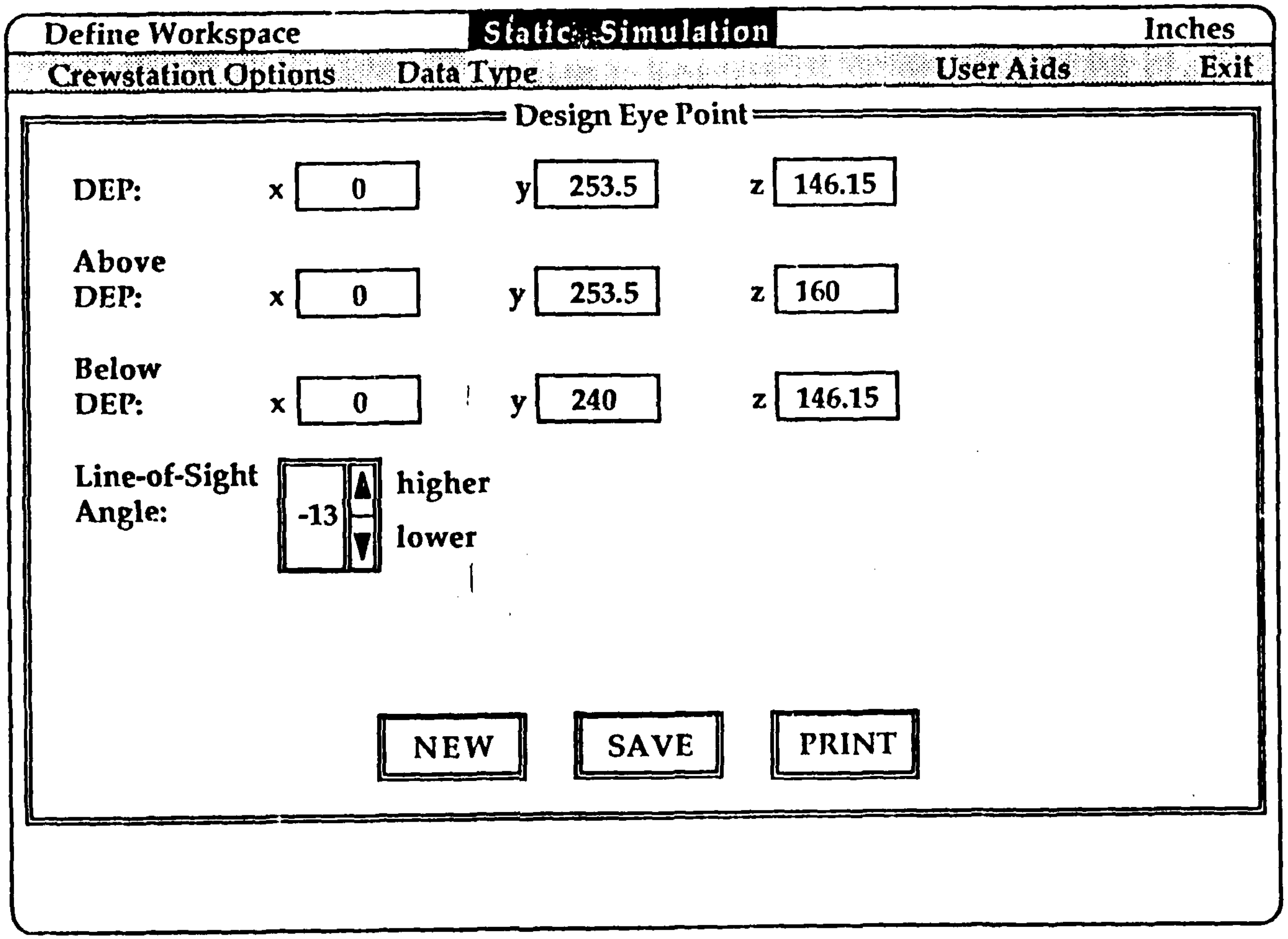
- **New** — clears all input fields and places the text cursor in the DEP x-coordinate field.
- **Save** — saves the current design eye point data as displayed on the screen. The following validation is performed prior to the actual saving the design eye point description:
  - The vector from the DEP-up location to the DEP must be perpendicular to the vector from the DEP-forward location to the DEP.
  - The LOS angle must be between -90 and 90 degrees.
- **Print** — print the current set of design eye point data on the printer.

The workspace design eye point dialogue window contains the following text entry boxes:

- **DEP** — entry of the x, y, and z coordinates of the design eye point.
- **Above DEP** — entry of the x, y, and z coordinates of a point directly above the design eye point.
- **Front DEP** — entry of the x, y, and z coordinates of a point directly in front of the design eye point.

The following click boxes are used in the workspace design eye point dialogue window:

- **Line-of-Sight Angle** — entry of the number of measurements included in the workspace as an integer number between -90 and 90 degrees.



Design Eye Point

DEP:	x 0	y 253.5	z 146.15
Above DEP:	x 0	y 253.5	z 160
Below DEP:	x 0	y 240	z 146.15

Line-of-Sight Angle: -13 ▲ higher ▼ lower

Figure 7-6: Workspace Design Eye Point.

The DEFWORK workspace seat window is used for entering the necessary information to describe the seat data. The seat data consists of a seat back angle, the seat pan angle, the harness location, the seat track and the points defining the seat. Four types of seat tracks can be defined:

- No track indicating that no movement of the seat is possible.
- Single track in a Down-Back — Down-Forward direction.
- Single track in a Down-Back — Up-Back direction.
- Double Track.

The seat is defined by the coordinates of the following points:

- Seat reference point down-back,
- Seat reference point up-back, and
- Seat reference point down-forward.

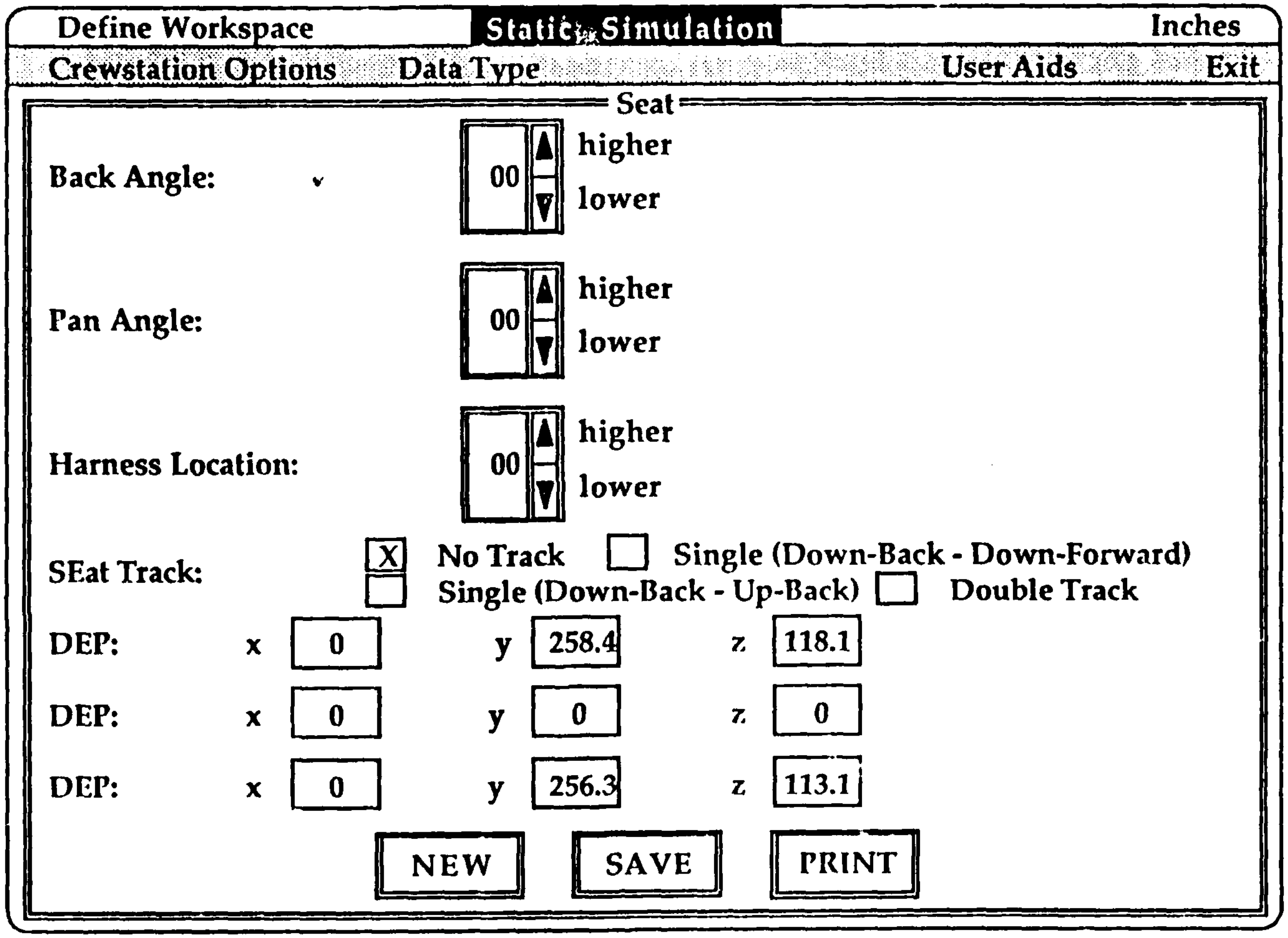
The harness percentage indicates the percentage of the distance from the midline to the shoulder joint at which the harness intersects the shoulder. In the STATSIM reach analysis, the harness percentage specified by the user only affects Zone 2 hand reaches for controls with a locked harness. A harness percentage of zero would indicate no harness effect and a harness percentage of 100 would indicate no shoulder movement. It is recommended that a harness condition of 50 percent be used if the harness restraint system is unknown.

The current activity area of the title bar will contain the words 'Define Workspace'. The status information area contains the word 'Inches'. The workspace seat window, as illustrated in Figure 7-7, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the workspace name field.
- **Save** — saves the current workspace seat as displayed on the screen. The following validation is performed prior to the actual saving the workspace seat:
  - The seat back angle must be between 0 and 90 degrees.
  - The seat pan angle must be between 0 and 90 degrees.
  - The seat pan angle must be less than the seat back angle.
  - The harness percentage must be between 0 and 100.
- **Print** — print the current workspace description on the printer.

The workspace seat dialogue window contains the following text entry boxes:

- **Seat (Up-Back)** — entry of the x, y, and z coordinates of the seat up-back point.
- **Seat (Down-Forward)** — entry of the x, y, and z coordinates of the seat down-forward point.



Seat

Back Angle: 00 ▲ higher ▼ lower

Pan Angle: 00 ▲ higher ▼ lower

Harness Location: 00 ▲ higher ▼ lower

SEat Track: ☒ No Track □ Single (Down-Back - Down-Forward)
□ Single (Down-Back - Up-Back) □ Double Track

DEP: x 0 y 258.4 z 118.1

DEP: x 0 y 0 z 0

DEP: x 0 y 256.3 z 113.1

Figure 7-7: Workspace Seat Screen.

- **Seat (Down-Back)** — entry of the x, y, and z coordinates of the seat down-back point.

The following check boxes are used in the workspace seat dialogue window:

- **Seat Track** — specify the seat track as no track, single (down-back — down forward), single (down-back — up-back), or double track.

The following click boxes are used in the workspace seat dialogue window:

- **Back Angle** — entry of the seat back angle as an integer number between 0 and 90.
- **Pan Angle** — entry of the seat pan angle as an integer number between 0 and 90.
- **Harness Location**— entry of the harness percentage as an integer number between 0 and 100.

The DEFWORK control window is used for entering control information for a workspace. A maximum of 50 controls may be defined for a workspace. The current activity area of the title bar will contain the words 'Define Workspace'. The status information area contains the word 'Inches'. The status information area of the title bar will also contain the sequential integer number of the control in the workspace currently being processed starting from one.

If the workspace is foot-anchored, only hand controls may be specified. The following control characteristics are specified for each control:

- Control Name.
- Control Description.
- Control Location as the x,y, and z coordinates within the workspace.
- Body part to be used to reach the control. The body part (left hand, right hand, both hands, left foot, right foot, both feet) specifies the body part the operator will use to reach the control. If both hands (or both feet) are specified, the reach analysis is performed for each body part and the results for the hand (or foot) placed furthest from the control are recorded.
- Hand grip to be used (for hand controls only). In a clenched grip, the control is grasped with the fingers wrapped around the control, such as a joystick. In a fingertip grip, the control is grasped with the fingertips, such as a knob or a toggle switch. An extended grip is used for controls such as a pushbutton.
- Harness condition (for hand controls only). If the harness is locked, only Zone 1 and 2 reaches will be attempted. If the harness is unlocked or if the control is operated by the foot, only Zone 3 reaches will be attempted.
- Critical control flag. The accommodation analysis reports the data on the number of operators able to reach controls in two groups — all controls or critical controls only.
- Adjustable limits for moveable controls. For adjustable controls, an additional position that together with the control location represents the limits of its

linear range of motion is required. In the reach analysis for adjustable controls, the operator attempts to reach each of the end points of the control location. The reach analysis reports the results for each end point separately and for the worst case of the two end points.

The specification of the body part and harness condition indicates the kind of reaches that will be attempted in the reach analysis. The three types of reaches are

- Zone 1 — the shoulder harness is locked and the operator does not strain against the harness. The lumbar, thoracic, interclavicular and clavicular links are immobile. The remaining links are allowed to move within their angular limits.
- Zone 2 — the shoulder harness is locked and the operator strains against the harness. The lumbar, thoracic, and interclavicular links are immobile. The clavicular link is allowed to move within the confines of the harness. The remaining links are allowed to move within their angular limits.
- Zone 3 — the shoulder harness is unlocked. All links are allowed to move within the bounds of their angular limits.

The workspace control window, as illustrated in Figure 7-8, contains the following *pushbuttons:*

- **New** — clears all input fields and places the text cursor in the control name field.
- **Save** — saves the control data as displayed on the screen for the current control. The following validation is performed prior to *the actual saving the* control information:

  The control name and description are valid names, i.e., it starts with an alphabetic character (a-z) and does not contain any illegal characters.

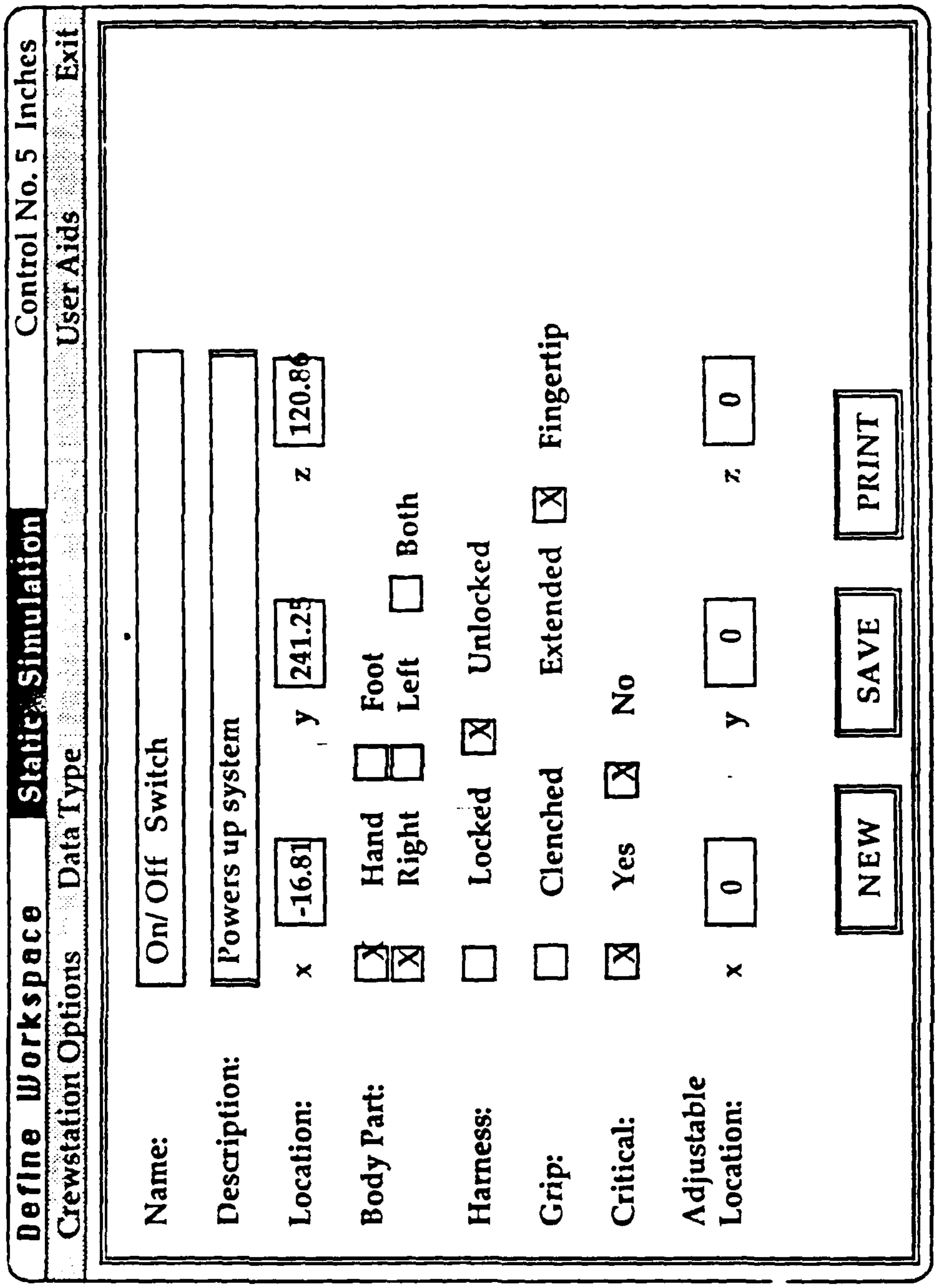
  When the save operation is completed, all fields in the dialogue window will be cleared and the text cursor placed in the control name field.
- **Print** — print the current set of workspace control data on the printer.

The workspace control dialogue window contains the following text entry boxes:

- **Control Name** — entry of the control name as a maximum of 28 characters.
- **Control Description** — entry of the workspace description as a maximum of 80 characters.
- **Control Location**— entry of the x, y, and z coordinates of the control location.
- **Adjustable Location**— entry of the x, y, and z coordinates of the adjustable control location.

The following check boxes are used in the workspace seat dialogue window:

- **Body Part** — specify the body part to be used for the control as hand or foot and then as right, left, or both.
- **Harness** — specify the harness as locked or unlocked.



Control No. 5 Inches

Name: On/ Off Switch

Description: Powers up system

Location: x -16.81 y 241.25 z 120.86

Body Part: ☒ Hand ☒ Right □ Foot □ Left □ Both

Harness: □ Locked ☒ Unlocked

Grip: □ Clenched □ Extended ☒ Fingertip

Critical: ☒ Yes □ No

Adjustable Location: x 0 y 0 z 0

Figure 7-8: Workspace Control Screen.

- **Grip** — specify the hand grip as clenched, extended, or fingertip.
- **Critical** — specify whether the control is judged critical to the successful operation of the workstation.

The DEFWORK workspace head clearance window is used for entering the necessary information to describe the head clearance data. The head clearance data consists of a minimum clearance distance, helmet thickness, and the coordinates of a point on the canopy to be used to test if the operator has sufficient head clearance. If these values are zero, no head clearance checks are performed by the accommodation analysis.

The current activity area of the title bar will contain the words 'Define Workspace'. The status information area contains the word 'Inches'. The workspace head clearance window, as illustrated in Figure 7-9, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the minimum clearance field.
- **Save** — saves the current head clearance as displayed on the screen. The following validation is performed prior to the actual saving the workspace seat:
  - The head clearance point must be above and in front of the seat.
  - The helmet thickness must be between 0 and 12 inches.
- **Print** — print the current workspace head clearance on the printer.

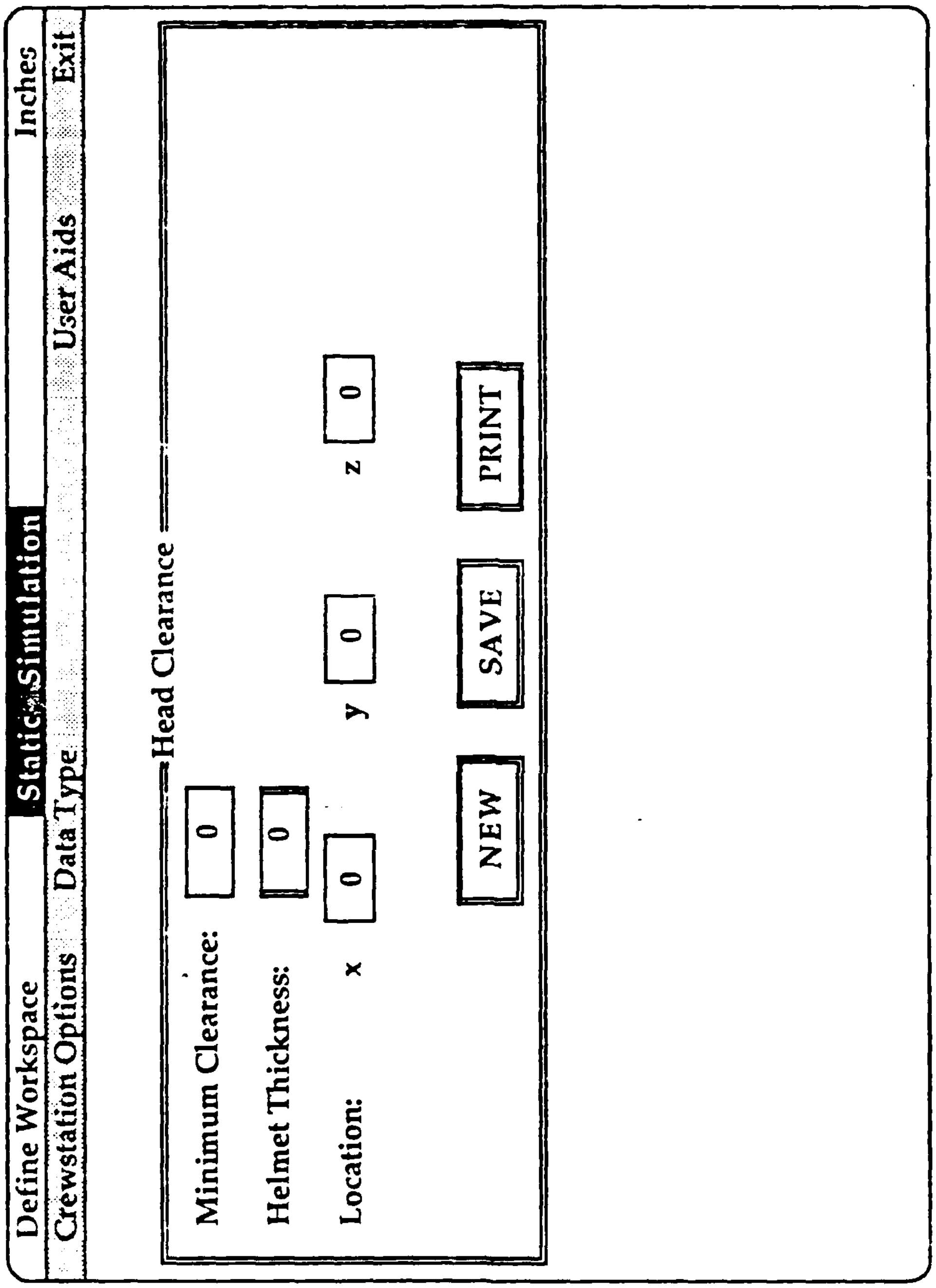
The workspace head clearance dialogue window contains the following text entry boxes:

- **Minimum Clearance** — entry of the head clearance minimum distance.
- **Helmet Thickness** — entry of the helmet thickness.
- **Clearance Location** — entry of the x, y, and z coordinates of the point to be cleared by the head.

7.1.3.3 ANALYZE

Description

The ANALYZE module of STATSIM performs the accommodation analysis. ANALYZE computes the position of each link as each of the operators in the sample reaches for each control. Prior to the control reach analysis, the effects of clothing are applied to each affected link, the harness effect is calculated, the base of the lumbar joint for seated operators is determined, and each operator is positioned at the anchorage point. The link-person is constructed link-by-link reaching for a specified point in the workspace. As each link is added to the previous link, an



Head Clearance

Minimum Clearance: 0

Helmet Thickness: 0

Location: x 0 y 0 z 0

Figure 7-9: Workspace Head Clearance Screen.

attempt is made to point the end of the new link towards the reach point. This process is constrained by the angular limits of motion for each link. ANALYZE positions each operator in the workspace and generates data on the following:

- Vision accommodation to DEP and line-of-sight,
- Head clearance, ano
- Ability to reach controls for Zones 1, 2, and/or Zone 3 reaches as appropriate.

For a more detailed description of the accommodation analysis algorithms, see the CAR-II manual (Harris, et al., 1980). The user must specify the sample and workspace tables to be used in the accommodation analysis. These tables must have been previously defined.

ANALYZE Screens

The ANALYZE screen contains a diagram of pushbuttons and arrows indicating the functional flow of steps invoked in defining and performing an accommodation analysis as illustrated in Figure 7-10.

ANALYZE Title Bar. The title bar of ANALYZE contains the words Static Simulation as the function name and Accommodation Analysis in the current activity area.

ANALYZE Menu Bar. ANALYZE contains the following menu options of the menu bar as illustrated in Figure 7-10:

- **User Aids** — provides the capabilities to view files and obtain help.
- **Exit** — terminates ANALYZE and returns the user to the Static Simulation main menu.

The **User Aids** pull-down menu contains the following commands:

- **Help** command that allow the user to obtain help windows about the ANALYZE module.
- **Print Tables** command that allows the user to obtain a copy of any of the STATSIM tables on the printer.
- **View Tables** command that allows the user to obtain a copy of any of the STATSIM tables on the screen.

ANALYZE Main Screen. The ANALYZE main screen contains the following pushbuttons:

- **Specify Sample**— allows the user to select the soldier sample table containing the anthropometric measurements that is to be analyzed.
- **Specify Workspace**— allows the user to select the workspace table containing the workspace configuration that is to be analyzed.

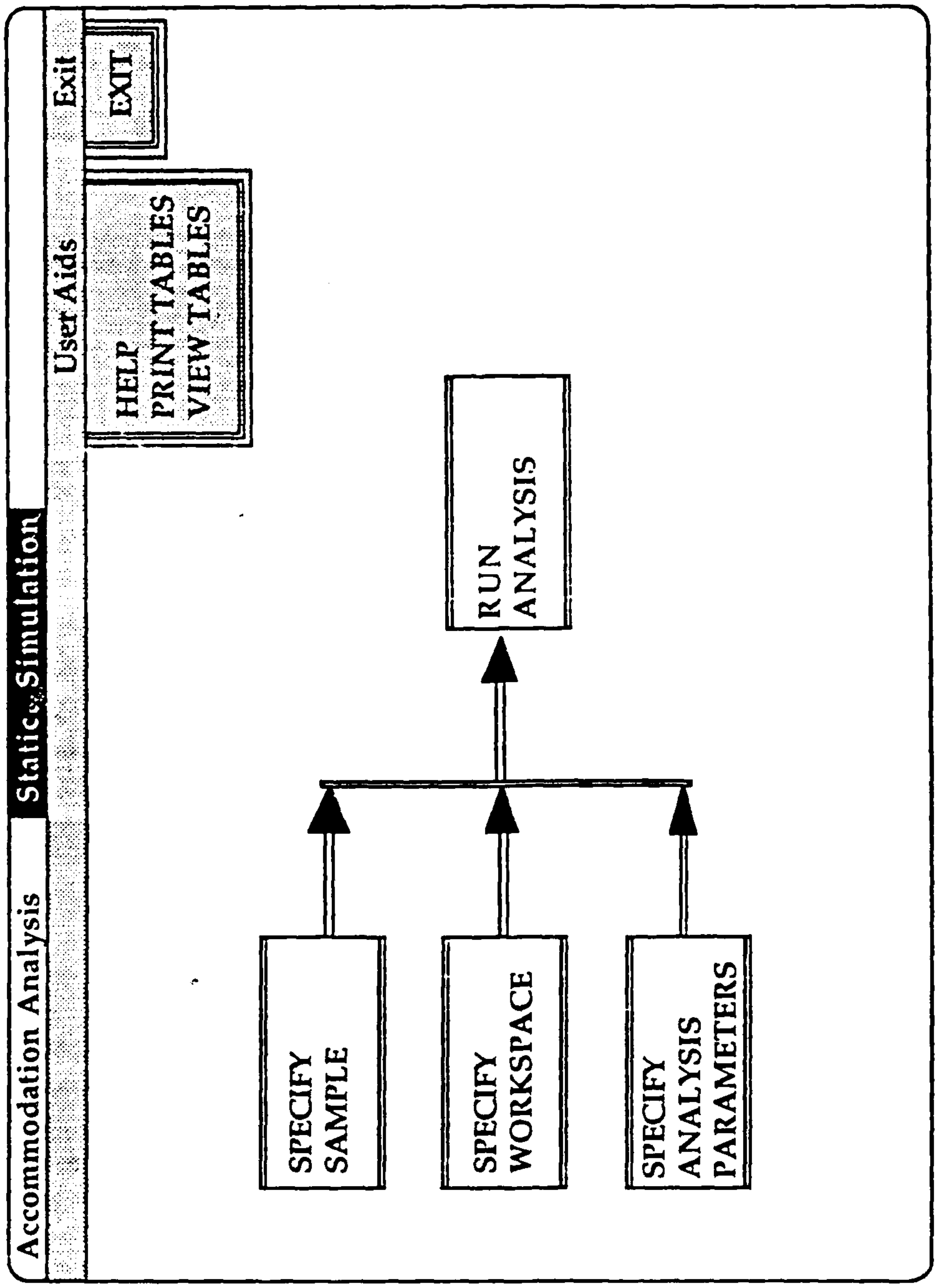


Figure 7-10: Accommodation Analysis Main Screen.

- **Specify Analysis Parameters**— allows the user to specify the accommodation analysis parameters.
- **Run Analysis** — runs the accommodation analysis.

Specify Sample Windows

The Specify Sample window is used for entering the necessary information to define the soldier sample to be used in the accommodation analysis and indicating which soldiers are to be included from the sample. The current activity area of the title bar will contain the words 'Analysis—Specify Sample'. The selected soldier sample file must have been previously generated using the GENSAMP module of HIG.

The Specify Sample window, as illustrated in Figure 7-11, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the soldiers field.
- **Save** — saves the selected sample parameters as displayed on the screen.
- **Print** — print the current sample parameters on the printer.

The Specify Sample window contains the following check boxes:

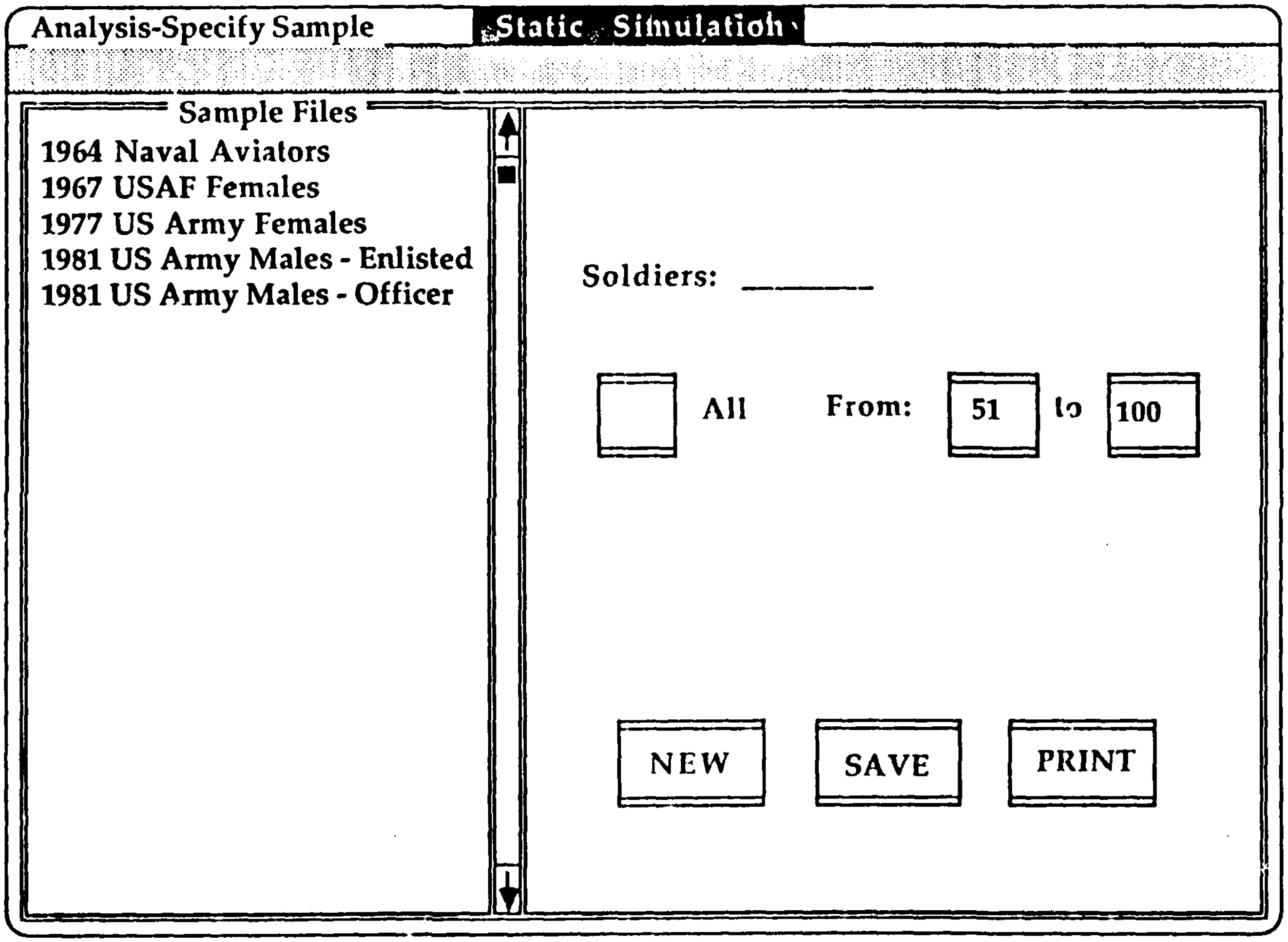
- **Soldiers** — the All checkbox indicates that all operators are to be included.in the analysis; the From checkbox indicates that only those operators in the range specified in the two text entry boxes are to be included.

A list selection box showing the list of available sample files is displayed in the upper left corner of the generate dialog window as shown in Figure 7-11.

Specify Workspace Window

The Specify Workspace window is used for entering the necessary information to define the soldier sample to be used in the accommodation analysis and indicating which soldiers are to be included from the sample. The current activity area of the title bar will contain the words 'Analysis—Specify Workspace'. The user can specify that all controls in the workspace are to be analyzed or specify the first and last control number in a range. The selected workspace file must have been previously generated using the DEFWORK module of STATSIM.

The Specify Workspace window, as illustrated in Figure 7-12, contains the following pushbuttons:



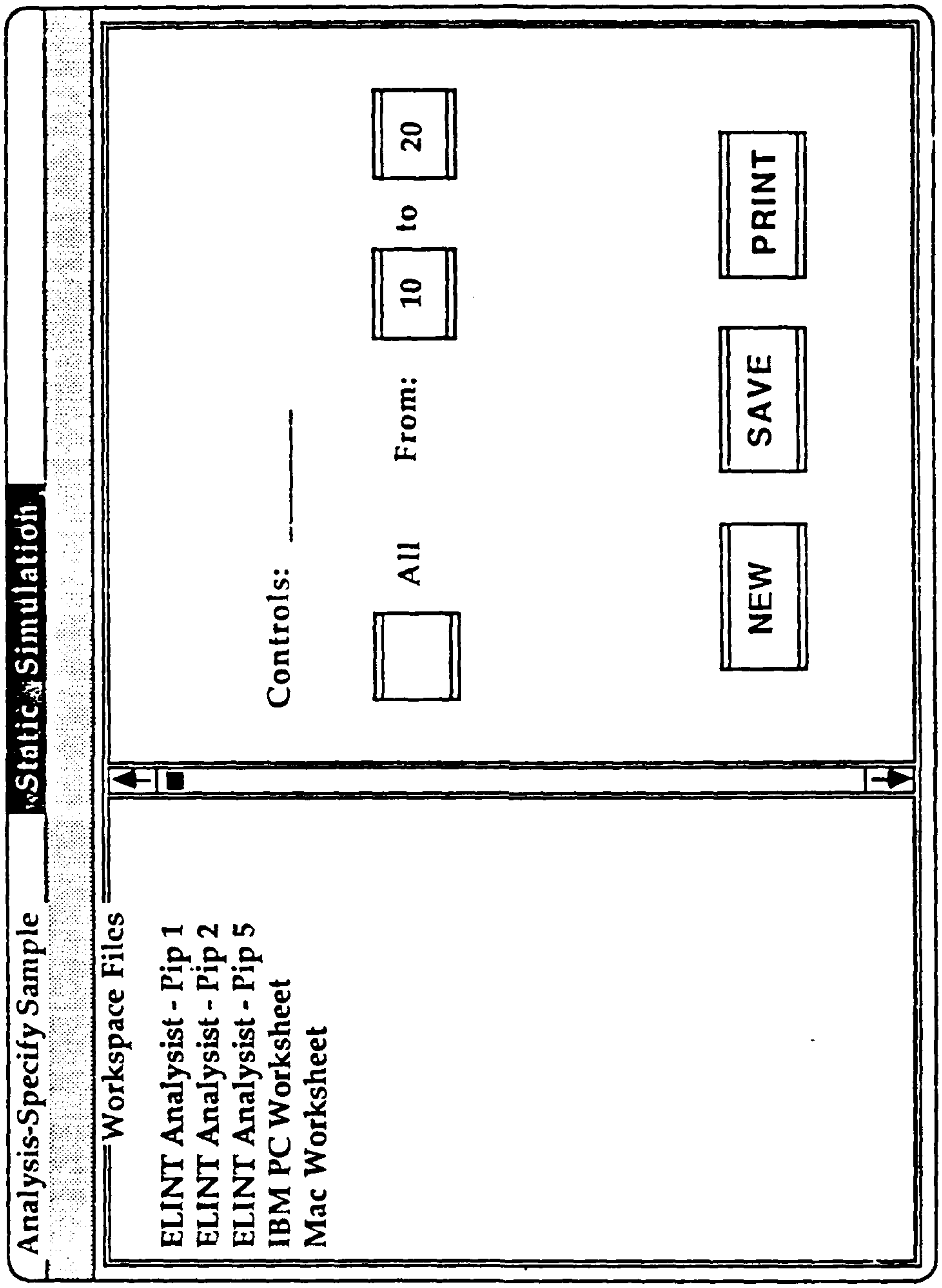
Sample Files

1964 Naval Aviators
1967 USAF Females
1977 US Army Females
1981 US Army Males - Enlisted
1981 US Army Males - Officer

Soldiers: ______

□ All From: 51 to 100

Figure 7-11: Specify Sample Source for Accommodation Analysis.



Workspace Files

ELINT Analysist - Pip 1
ELINT Analysist - Pip 2
ELINT Analysist - Pip 5
IBM PC Worksheet
Mac Worksheet

Controls: ______

□ All From: 10 to 20

Figure 7-12: Specify Workspace.

- **New** — clears all input fields and places the text cursor in the controls field.
- **Save** — saves the selected workspace parameters as displayed on the screen.
- **Print** — print the current workspace parameters on the printer.

The Specify Workspace window contains the following check boxes:

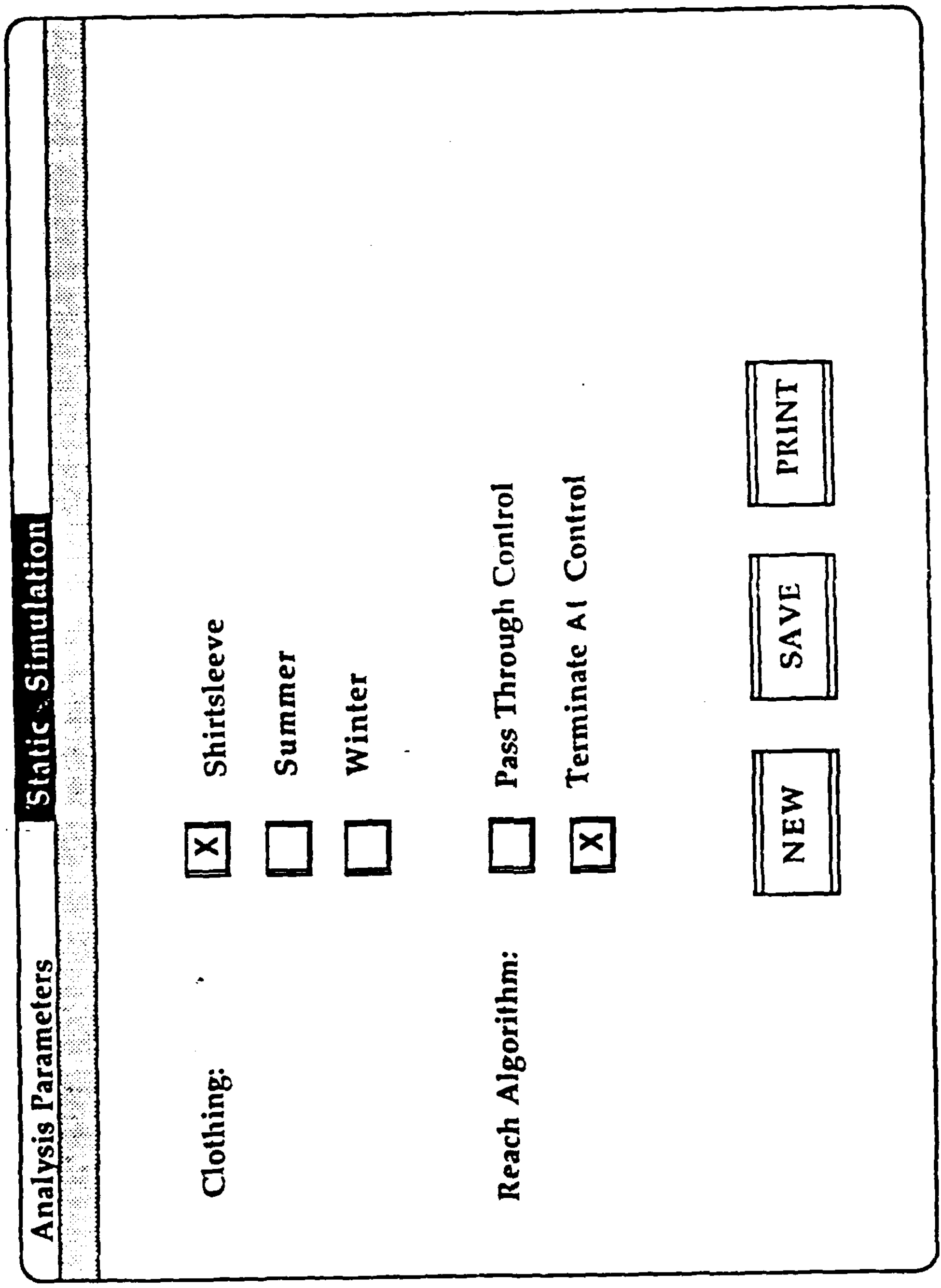
- **Controls** — the All checkbox indicates that all controls are to be included.in the analysis; the From checkbox indicates that only those controls in the range specified in the two text entry boxes are to be included.

A list selection box showing the list of available workspace files is displayed in the upper left corner of the generate dialog window as shown in Figure 7-11.

Specify Analysis Parameters Window

The Specify Analysis Parameters window is used specify the analysis parameters. STATSIM allows the user to specify that the operators arewearing"shirt-sleeve" clothing or flight clothing (summer or winter). The clothing specification modifies the appropriate link lengths and their angular limits of motion. "Shirt-sleeve" clothing specification indicates that no clothing effects are applied. The specified clothing option will be applied to all operators included in the accommodation analysis.

STATSIM also allows the user to select the reach algorithm to be utilized in determining if an operator can reach a control. The "pass through" algorithm considers a control "reachable" if any part of the arm or leg passes through the control location. With this algorithm, it is not necessary for the end of the hand or foot to be placed precisely at the control location in order for a reach to be achieved. This definition of "reachability" is useful since STATSIM's only concern is whether a point is reachable and not how the reach is performed; it is assumed that if any part of the arm (leg) passes through a point, the arm (leg) may be configured so that the hand (foot) terminates at that point. The assumption of "pass through control" does not hold true for all cases, but it considerably simplifies the STATSIM reach algorithm and is extremely reliable for controls generally encountered in workspace designs (i.e., controls in front of and not very close to the operator). However, in order to analyze how a reach is performed and to generate reach envelopes, it was necessary to modify the algorithms to attempt to actually place the end of the hand (foot) at the control location. This "Terminate at Control" algorithm is more reliable for predicting reaches for controls behind or close to the operator. This algorithm greatly increases the executing time to perform a STATSIM accommodation analysis so the user can select the desired reach algorithm by specifying the reach algorithm parameter.



Clothing: ☒ Shirtsleeve □ Summer □ Winter

Reach Algorithm: □ Pass Through Control ☒ Terminate At Control

Figure 7-13: Select Conditions Screen.

The current activity area of the title bar will contain the words 'Analysis—Specify Analysis Parameters.' The Specify Analysis Parameters window, as illustrated in Figure 7-13, contains the following pushbuttons:

- **New** — clears all input fields and places the text cursor in the clothing field.
- **Save** — saves the selected analysis parameters as displayed on the screen.
- **Print** — print the current analysis parameters on the printer.

The Analysis Parameters window contains the following check boxes:

- **Clothing** — specifies the clothing option as shirt-sleeve, summer, or winter.
- **Reach Algorithm** — specifies the reach algorithm as pass through control or terminate at control.

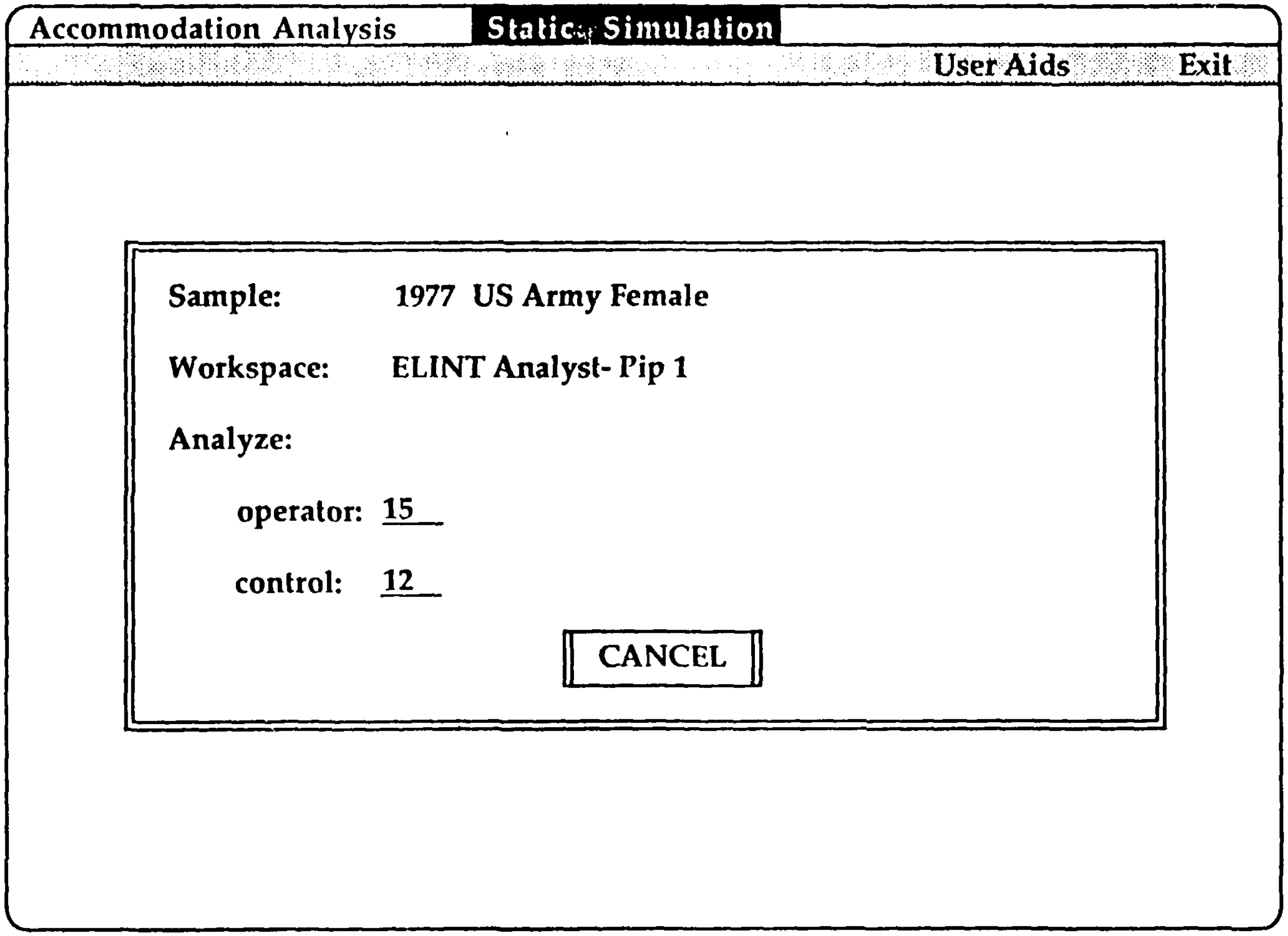
Run Analysis Message Window.

When the user depresses the run analysis pushbutton, the Run Analysis message window shown in Figure 7-14 is displayed. This window indicates the name of the selected sample and workspace tables and indicate which member of the sample and control are currently being analyzed. It contains a Cancel pushbutton that allows the user to terminate the accommodation analysis process.

7.1.3.4 RESULTS

The RESULTS module is used to examine data produced by running the accommodation analysis. It contains a series of standard reports that are used to assist in the analysis of the accommodation results. The available reports are the following:

- Parameters — displays information about the soldier sample, workstation geometry, and specified analysis parameters.
- Anchorage accommodation — displays information about the ability of the operators to achieve the specified anchorage.
- Vision accommodation — displays information about the ability of the operators to achieve vision including
    - Percentage of operators positioned to the DEP.
    - Percentage of operators positioned above, on, and below the LOS for those positioned both forward and behind the DEP. Mean distance of the operator's eye from the DEP is also displayed for the above combinations.
    - The average vertical and horizontal displacement of the operator's eye from the DEP.
- Control Reach Summary — presents the results of the reach analysis for each control for the appropriate zone(s). The control name, body part, harness



Sample: 1977 US Army Female

Workspace: ELINT Analyst- Pip 1

Analyze:

operator: 15

control: 12

Figure 7-14: Static Simulation Analysis Screen.

condition and grip type for hand controls are shown for each control. The percentage of the sample accommodated is shown for each of the appropriate reach zones. The average and worse case X, Y, and Z components of the distance and the actual distance between the end of the last link and the control is reported. If any operators were unable to reach controls because of angular limit problems, the average and worse case data are reported separately.

Results Screens

RESULTS Title Bar. The Results screen consists of a title bar, a menu bar, a text viewing window with a scrollbar, and a number of dialog boxes used for program interaction with the user. The title bar contains the words 'Static Simulation' as the function name in the center. The current activity area contains the words 'Results'.

Results Menu Bar. The menu bar for the Results module contains the following menu options:

- **Report Type** — selection of the desired report.
- **User Aids** — provides the capabilities to view files and obtain help.
- **Exit** — terminates SIMMAIN and returns the user to the MMP6 main menu.

The **Report Type** pull-down menu contains the following commands:

- **Parameters** — opens the Parameter report and displays the contents in a text viewing window.
- **Anchorage accommodation** — opens the Anchorage accommodation report and displays the contents in a text viewing window.
- **Vision accommodation** — opens the Vision accommodation report and displays the contents in a text viewing window.
- **Control Reach Summary** — opens the control reach summary report and displays the contents in a text viewing window.

The **User Aids** pull-down menu contains the following commands:

- **Help** command that allow the user to obtain help windows about the RESULTS module
- **Print Tables** command that allows the user to obtain a copy of any of the STATSIM tables on the printer.
- **View Tables** command that allows the user to obtain a copy of any of the STATSIM tables on the screen.

Results Text Viewing Window. The text viewing window consists of a text area and a scrollbar. The text area is 23 rows by 7 columns. The scrollbar is to the right of the text area and has four control buttons and a relative file position indicator. The controls on the scroll bar allow the user to scroll up and down a line or a page at a time.

8. SUMMARIZE AND ANALYZE

Once the dynamic simulation has been run, all that is left to do is summarize and analyze the results. The summary module does the summary function. The analysis evaluates the summarized results in the context of mission requirements.

8.1 Summary Module

The purpose of this module is to summarize and reorganize the results in terms of UIPs and KUIPs coming out of the dynamic simulation. A dynamic simulation is written using RULES, ACTIONS, and OBJECTS which have one or more CHARACTERISTICS. The relations among and between these vocabulary tools of the simulation language are illustrated in Figure 8-1. In particular, the prototypic UIP and KUIP will have one RULE, two ACTIONS, and one OBJECT. The importance of this relation lies in the fact that summarizing can be performed entirely in the context of rules.

The summary module is invoked off the Top Level screen and only has one screen specifically associated with it; that of view results (e.g. Figure 6-13). All of the results are manipulated in R:BASE form using the R:BASE procedures. A results file is created as illustrated schematically in Table 8-1. The time and accuracy entries represent the average values from the tables. These values are then modified by the analysis routine to represent the values associated with performance for the subject currently being analyzed.

8.2 Analysis Module

The purpose of this module is to evaluate the capability of of each of the hypothetical individuals to perform each of the critical tasks and the entire set of critical tasks. By doing both the individual tasks and the entire set of tasks as a whole, the analyst can evaluate specific personnel characteristics. Alternately, he can evaluate system performance against a selected subset of the personnel characteristic measures. That is, he may not want to use any values except those generated from ASVAB scores.

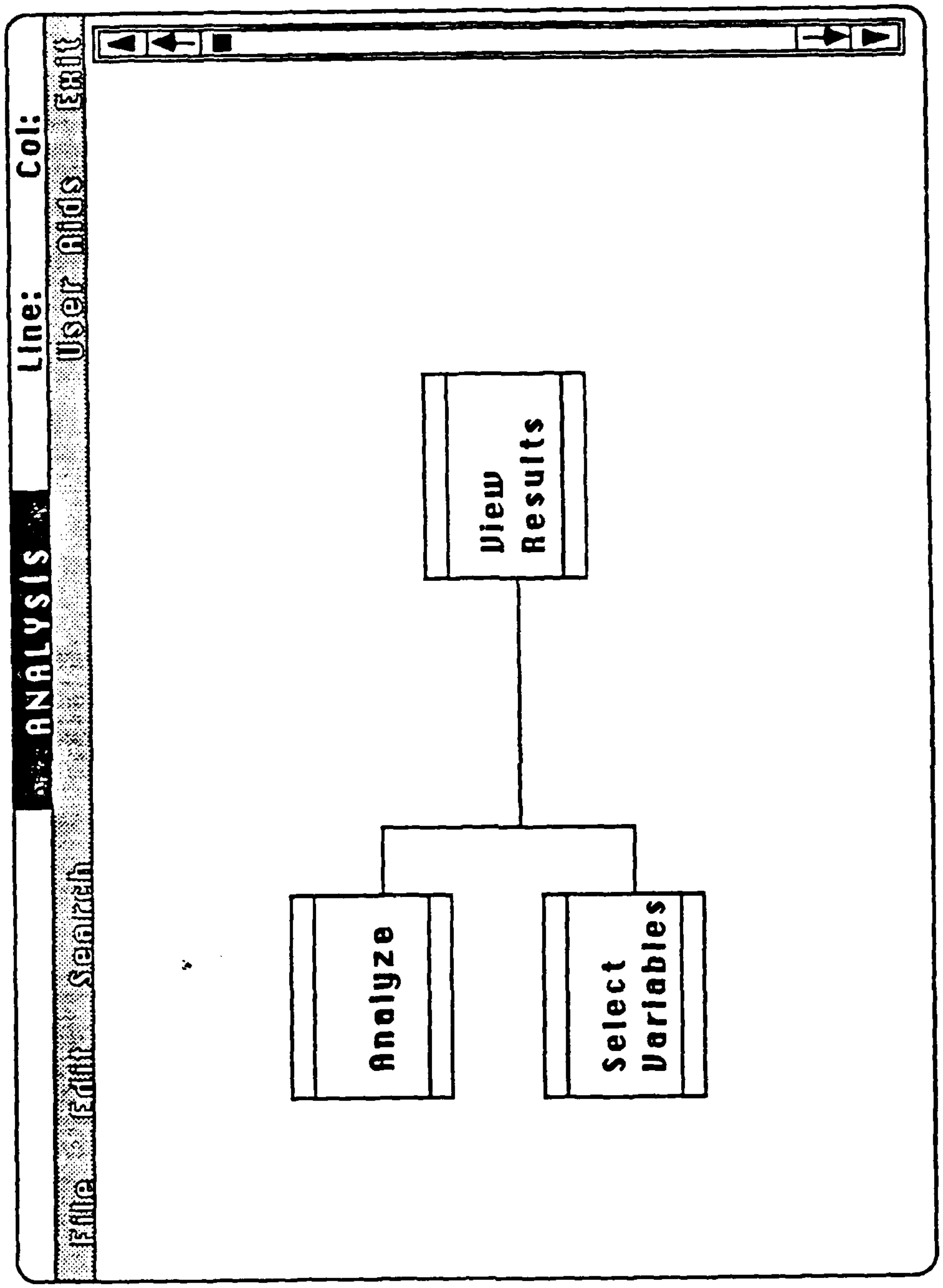


Figure 8-1: Analysis Screen Showing Options.

Table 8-1: Illustration of a Potential Outcome of a Simulation Organized by Task, Rules, UIPs and KUIPs.

Task	Rule	#	KUIP	Time	Acc.
Task 1					
		1	$KUIP_1$	120	90
		4	$KUIP_2$	350	86
		2	UIP_4	195	76
		2	UIP_3	75	98
		1	$KUIP_7$	120	90
		4	$KUIP_6$	350	86
		2	UIP_5	175	98
		4	$KUIP_{22}$	320	81
		2	UIP_6	75	98
Task 2					
		4	$KUIP_{12}$	40	65
		2	UIP_3	75	98
		2	UIP_9	330	75
		3	$KUIP_6$	400	87
		2	UIP_1	185	98
		2	UIP_{12}	175	93
		1	UIP_{21}	220	95
.					
.					
.					
Task N					

The theory behind the module is to combine the KUIP tables (generated by the KUIP module described in Section 5) for personnel characteristics, the performance criteria from MMP1 and the performance evaluations coming from the dynamic simulation and summarized by SUMMARY. The performance on critical tasks for the average hypothetical individual is adjusted in terms of performance capabilities expected for each hypothetical individual and then evaluated against the criterion for system performance. These evaluations are done in terms of success or failure (0's and 1's for computer purposes).

The module develops three separate evaluations:

- The proportion of hypothetical individuals who accomplish each of the critical tasks – the detail is evaluated. This will indicate tne proportion of individuals who can accomplish an individual critical task.
- The proportion of hypothetical individuals who accomplish all of the critical tasks – the overall summary is evaluated. This will indicate the proportion of soldiers who can accomplish the entire set of tasks.
- The proportion of hypothetical individuals who can accomplish the tasks based on a subset of the *personnel characteristics*. This can be used to reduce the number of personnel characteristic variables used to develop the analysis.

The analysis is accomplished in two steps. The first step is to calculate an individual's performance and compare the performance to the criteria. The results of this analysis are written to a file. The second step provide the summaries of the results according to the descriptions above.

Having reached this point, the remaining analysis requires some fairly simple mathematics. Performance for a specific hypothetical individual is calculated in the following way:

- The KUIP transformation table (KUIPFREQ) which was discussed in Section 5 contains the frequencies of usage of KUIPs and mean time for each of the test variables for the personnel characteristics database.
- Use of this table in a "best fit" analysis also provides a check on the viability of the KUIP values generated for the performance tests.
- The inverse of the KUIPFREQ matrix is multiplied by the personnel characteristic scores for the HI. This yields the KUIP values for this HI. (Actually, a pseudo inverse calculation will be probably used to permit 'conditioning' of the values, since neither time nor accuracy can be negative.)

  $HI(i,j) = V(I) * KUIP^{-1}$
- The HI values are substituted in the simulation results table, replacing the values for the 'average' hypothetical individual.
- These new values are then compared against the criterion values and the results of this comparison are entered into the results table.
- The following probability functions are then calculated using the results table.

PT = Proportion of critical tasks completed.

NT = number of critical tasks.

q_1 = the proportion who accomplished all tasks.

NCASES = number of hypothetical individuals.

The proportion of hypothetical individuals who accomplish individual tasks is arrived at by:

$$PT(T) = [C(1,T,1) + C(2,T,1) + ... + C(NCASES,T,1)] / NCASES$$

for each of the tasks. Thus, each PT represents a proportion of completion for each of the NT tasks (summed across hypothetical individuals and divided by the number of hypothetical individuals).

The proportion of hypothetical individuals who complete all critical tasks is a more stringent criterion and is given by:

$$q_1 = [P(1) + P(2) + ... + P(NCASES) / NCASES$$

where each P is either a 1 or a 0, indicating success or failure.

A discriminate function can be used to evaluate the a subset (reduced range of values) of personnel characteristic values in terms of completing the critical tasks.

$$Pr(1|x) = q_1 * p_1(x) / [q_0 * p_0(x) + q_1 * p_1(x)]$$

where $P_1(x)$ and $p_0(x)$ may be estimated by:

$$P_i(x) = C_i * \exp[-1/2 * (x - m_i)' * S_i^{-1} * (x - m_i)]$$

for i = 0 and i = 1. In this formula, m_i is the for the sample element of the subset of personnel characteristics and S_i^{-1} is the inverse of the covariance matrix of the subset of personnel characteristic values (Anderson, 1984; Colley & Lohnes, 1971).

Output

The output is entered into the file table corresponding to that created by the hypothetical individual generation routine described in Section 5. The two files are linked and indexed by subject number. Accordingly, a full report may be obtained, detailing specific individual personnel characteristics and criterion performance or a summary of the results can be obtained.

View Results

The view results screen is similar to that shown in Figure 6-13.

9. GENERAL PURPOSE TOOLS

There are several pieces of software which will be purchased to implement SEPTA. Skylights will be used to generate the screens and R:BASE will be used as a database. R:BASE contains several important features which permit it to be used as an editor and report generator as well as an organizer of data files. These products are discussed in this section.

9.1 Skylights/GX Usage

Skylights is a commercially available piece of software which is used as a development tool. While the products (screens) developed with Skylights are included in SEPTA, Skylights itself is not included.

Skylights provides two window editors — one alphanumeric and the other graphic for building windows and defining "touch zones." The graphics editor will not be used in this implementation of SEPTA. The user interface will be written using alphanumeric windows because the speed of execution is much greater. Touch zones are areas on the screen that when the mouse cursor moves over them or the user clicks on them cause the associated function defined in the window editor (demon) to be executed. This allows the development of an event driven user interface that is necessary for a menu/mouse driven program. The alphanumeric window editors are used in conjunction with a extensive library of C callable routines to develop the user interface described using Skylights.

A key feature of Skylight is the ability to define interactively various screen components and store them in a library. With other packages and tool kits, common front end structures and visuals, such as windows, menus, icons, boxes, and screen captions, are often implemented dynamically in the application program. Skylights uses a different approach. Most of the necessary structures and visuals are created interactively with the Skylights editor and saved in separate window catalog files. Low-level functions for screen handling, drawing boxes, color definition, etc. are available.

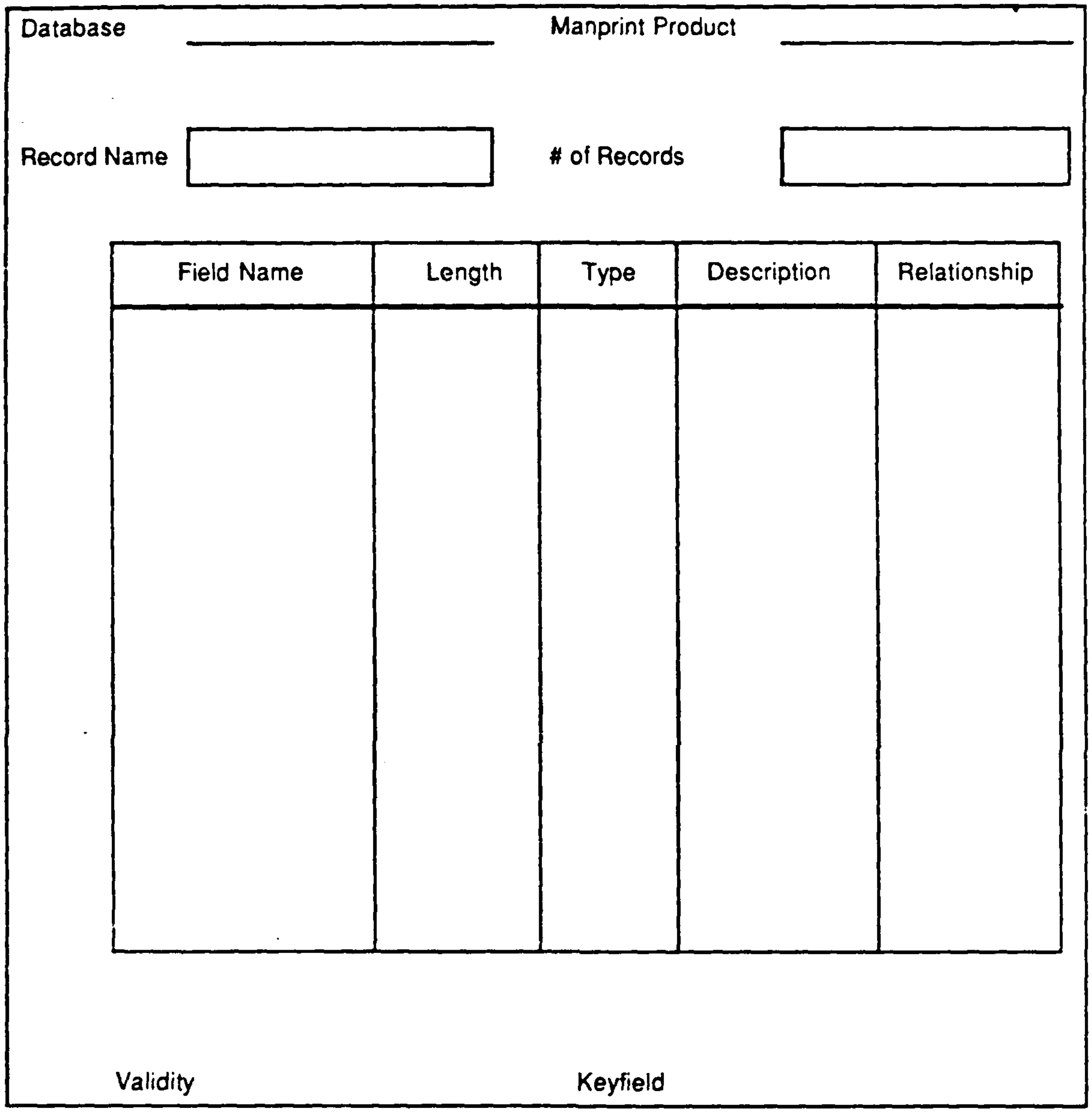
The Skylights editor is used to create windows. A window is a fragment of the screen which has a picture and imbedded active areas (touch zones), menus, icons, etc. Windows are saved in a file called a window catalog. The defined windows are loaded into an application at run time using library functions calls.

Using a pointing device, a window is defined, character graphics are drawn to create the screen display and shape, touch zones are specified along with how the application should respond to the touch event. In addition, optional properties such as audio feedback, video feedback, and the name of the routine responsible for processing touch events in the zone ("demons") can be specified. The run time libraries contain window management functions, a touch events handler, as well as various screen, keyboard, and speaker handling functions.

9.2 Foreign Data and R:BASE

There was no widely accepted database among the contractors, and accordingly we made our own selection. We will use R:BASE System V produced by Microrim, Inc. as a portion of MMP6 to manage and organize the necessary data files. The advantage of this is the prebuilt features available such as indexing, ease in changing file contents, etc. Because this is a commercially available product and one on the GSA list, we will not go into great detail on the design. We will simply highlight some of the features which lead to the selection.

- C language interface. A program library is available which provides routines which allow access to the data base files from our own programs.
- Contains FileGateway which allows importing and exporting data in a variety of formats, including DIF and ASCI.
- Graphics capability which can access the data files. While this graphic capability is not planned as a feature with SEPTA, having the capability available is a bonus.
- Up to 80 files may be incorporated in a single database, and multiple databases are possible.
- Entry forms can be created easily for data entry and editing.
- R:BASE includes extensive reporting capabilities which will permit the user to define additional reports easily as appropriate.
- Run-time versions can be provided which is a consideration for the distribution of SEPTA.
- Data file password capability. Some files will not require changes very often, such as correlation matrices. These will be coded to read only status for routine work. When changes have to be made, the user will be required to enter a password.



Database ____________________ Manprint Product ____________________

Record Name □ # of Records □

Field Name	Length	Type	Description	Relationship

Validity Keyfield

Figure 9-1: The Form Used to Define a Record and Fields within a Particular Database File.

9.2.1 Data Imported from Other Sources

The agreement and overall specification for all MMP products is to use the Data Interchange Format (DIF) for data to be exported and imported from other products. Accordingly, these can be entered into R:BASE without difficulty and will involve little effort on the part of the user. The data written out by products 1 and 5 will be consistent in form and we will provide the necessary file structure to read them in.

9.2.2 Data Internal to SEPTA

Data created within SEPTA will go through R:BASE. The reason for this is to facilitate maintenance and possible later improvements. The facilities offered by R:BASE are convenient and make files consistent as to headers, etc.

9.2.3 New Data Entered Into SEPTA

Before the user enters new data, he will have to indicate the type of data and the module the data will "belong" to as well as the database he is working on. The User screens will guide the selection of the appropriate file names. Because we will be using R:BASE, the format will be specified from within SEPTA and will be acceptable to the programs. That is, the user will be guided; if he is entering a correlation matrix, he will be required to enter numbers. The specifics of entering data depends heavily on the module and the type of data being entered. As each module is discussed, it will be clear as what the specifics are. In general, Figure 9-1 shows definitions which will be required.

9.3 Data Editor

A data editor will be built for SEPTA using the routines provided with R:BASE. The principal use of the data editor is to facilitate changing data coming in from other modules. There is no way to insure the data will be appropriate in all aspects or that the terminology will always be consistent. For example, times and accuracies may be missing. SEPTA has default options to allow missing data to go through, but the user may wish to enter values, especially if the missing values are detected at key points. The database manager will be used so that the file formats will not have to be a concern of the user. The Data Editor will allow the user to edit his data files in order for them to be in the correct format for a program module as defined by R:BASE.

9.4 Report Generator (RG).

The report generator will be interfaced into SEPTA using R:BASE features and software calls. It is designed to allow screen viewing of any part of the data on the screen and mark the beginning and end of the desired report for printing. Thus, the user will be able to have a complete report or a partial report at his choosing.

10. REFERENCES

Adams, J. A. (1987). Historical review and appraisal of research on the learning, retention, and transfer of human motor skills. Psychological Bulletin, 101, 41-74.

Anderson, T. W. (1984). An introduction to multivariate statistical analysis. New York: Wiley.

Anthropometric Source Book. (1978). Volume II: A Handbook of Anthropometric Data. NASA Reference Publication 1024.

Bennett, J., Harris, R., & Stokes, J. (1982). Crewstation Assessment of Reach (CAR) Validation. (Technical Report 1400.21B). Willow Grove, PA: Analytics, Inc.

Bittner, A. C., Jr. (1976). Computerized accommodated percentage evaluation: Review and prospectus. Proceedings of the 6th Congress of the International Ergonomics Association. (Also published as Pacific Missile Test Center TP-76-46 (NTIS AD-A035205/4ST)).

Colley, W. W. & Lohnes, P. R. (1971). Multivariate data analysis. New York: Wiley.

Fleishman, E. (1967). Performance assessment based on an empirically derived task taxonomy. Human Factors, 9, 349-366.

Fleishman, E. (1972). On the relation between abilities learning and human performance. American Psychologist, 27, 1017-1032.

Guilford, J. (1967). The nature of human intelligence. New York: McGraw-Hill.

Guilford, J., & Hoepfner, R. (1971). The analysis of intelligence. New York: McGraw-Hill.

Harris, R., Bennett, J., & Dow, L. (1980). CAR-II – A Revised Model for Crewstation Assessment of Reach (Technical Report 1400.06B). Willow Grove, PA: Analytics, Inc.

Harris, R. & Iavecchia, H. (1984). Crewstation Assessment of Reach Revision IV (CAR) User's Guide (Technical Report 1800.10A). Willow Grove, PA: Analytics, Inc.

Harris, R. L. Sr. & Glover, B. (1984). Effects of digital altimetry on pilot workload. Paper presented at the the 1984 SAE Aerospace Congress and Exposition.

Iavecchia, H. & Harris, R. (1984). Incorporation of Female Link Transformations into CAR-IV (Technical Memorandum 1800.10). Willow Grove, PA: Analytics, Inc.

Iavecchia, H., Harris, R., & Rothenheber, E. (1986). CAR-IV Validation Study Results (Technical Report 1800.33B). Willow Grove, PA: Analytics, Inc.

Kaplan, J. D. & Crooks, W. H. (1980). A concept for developing human performance specifications. (Technical Memorandum 7-80). (Report No. PTR-2020-80-3).

Kennedy, K.W. (1978). Reach Capability of Men and Women: A Three-Dimensional Analysis (Technical Report AMRL-TR-7). Wright-Patterson AFB, Ohio: Aerospace Medical Research Laboratory.

Marquardt, L. & McCormick, E. (1974). The job dimensions underlying the elements of the Positions Analysis Questionnaire (PAW) (Form B). West Lafayette, IN: Purdue University, Occupational Research Center.

McCormick, E. (1974). The application of structured job analysis information based on the Position Analysis Questionnaire (PAQ). West Lafayette, IN: Purdue University, Occupational Research Center.

McCraken, J. H. & Aldrich, T. B. (1984). Analysis of selected LHX mission functions: Implications for operator workload and system automation goals. (TNA ASI479-24-84). Fort Rucker, AL: Anacapa Sciences, Inc.

Sticha, P. J. (1987). Models of procedural control for human performance simulation. Human Factors, 29, 421-432.

Trotter, M., & Gleser, G. (1952). Estimation of Stature from Long Bone Lengths of American Whites and Negroes. American Journal of Physical Anthropology, 10(6).

Wherry, R. Jr. (1985). Theoretical developments for identifying underlying internal processes. Volume 2: Modifications to hierarchical factor analysis: Positive Manifold (POSMAN) rotations (Technical Report 1800.31-TR-02). Willow Grove, PA: Analytics, Inc.

Wherry, R., Jr. (1986). Theoretical development of identifying underlying internal processes. Vol 1. The theory of underlying internal processes. (NAMRL Special Report 86-1 and NADC Report 86105-60). Warminster, PA: Naval Air Development Center.

Wherry, R., Jr. (1987). The Tactical Display Assessment Task Simulation (TACDATS) Model (Technical Report 2100.03-TR-02). Willow Grove, PA: Analytics, Inc.

Wherry, R., Sr. (1984). Contributions to correlational analysis. Orlando, FL: Academic Press.

11. BIBLIOGRAPHY

Adams, J. A. (1987). Historical review and appraisal of research on the learning, retention, and transfer of human motor skills. Psychological Bulletin, 101, 41-74.

Allen, J. A., Hays, R. A., & Buffardi, L. C. (1986). Maintenance training simulator fidelity and individual differences in transfer of training. Human Factors, 28, 497-509.

Analytics, Inc. (1975). Development of a quantitative display reading model for HOS Part I (Technical Report 1117-F). Willow Grove, PA: Analytics, Inc.

Analytics, Inc. (1975). Development of a quantitative display reading model for HOS Part II (Technical Report 1117-G). Willow Grove, PA: Analytics, Inc.

Analytics, Inc. (1982). HOPROC Assembler/Loader (HAL) user's/programmer's guide (Technical Report 1400.22A). Willow Grove. PA: Analytics, Inc.

Anderson, J. R. (1981). Cognitive skills and their acquisition. Hillsdale, NJ: Erlbaum.

Anderson, J. R. (1982). Acquisiton of cognitive skill. Psychological Review 9, 369-406.

Angus, R. G. & Heslegrave, R. J. (1984). The effects of sleep loss and sustained mental work: Implications for command and control performance. Downsview, Ontario, Canada: Defense and Civil Institute of Environmental Medicine.

Ayoub, M., Mital, A., Asfour, S., & Bethea, J. (1980a). Review, evaluation, and comparison of models for predicting lifting capacity. Human Factors, 22(3), 271-283.

Ayoub, M., Mital, A., Bakken, G., Asfour, S., & Bethea, J. (1980b). Development of strength and capacity norms for manual materials handling activities: The state of the art. Human Factors. 22(3), 271-283.

Banowetz, V., & Iavecchia, H. (1981). A human operator simulator model of the advanced signal processor (DICASS Processing) (Technical Report 1400.09). Willow Grove, PA: Analytics, Inc.

Banowetz, V., Iavecchia, H., & Berstrasser, J. (1981). Application of human operator simulator to Sensor Station 3 for the P3-C mod scenario (Technical Report 1400.16A). Willow Grove, PA: Analytics, Inc.

Baumann, J., & Williams, D. (1984). Design and development of a man-in-the-loop simulator for small tactical missiles. Proceedings of the Summer Computer Simulation Conference. Boston, MA.

Barrow, H. G. & Tenenbaum, J. M. (1986). Computational approaches to vision. In K. R. Boff, L. Kaufman, & J. P. Thomas (Eds.), Handbook of Perception and Human Performance. Vol. 2. New York: Wiley.

Belenky, G. (1986). Sustaining and enhancing individual and unit performance in continuous operations. Proceedings of the Soldier Performance Research and Analysis Review. Ft. Belvoir, VA.

Bittner, A., Jr. (1976). Computerized Accommodated Percentage Evaluation (CAPE): Review and prospectus (Technical Report TP-76-46). Point Mugu, CA: Naval Missile Center.

Bittner, A., Jr., Wherry, R., Jr., Glenn, F., & Harris, R. (1986). CADRE: A family of manikins for workstation design (Analytics Technical Report 2100.07B). Willow Grove, PA: Analytics, Inc..

Bittner, A., Jr., Wherry, R., Jr., Glenn, F., & Harris, R. (1987). CADRE: A family of manikins for workstation design. Paper presented to the Industrial Ergonomics and Safety Conference '87, Miami, FL.

Campbell, J. (1986). When the textbook goes operational. Paper presented at American Psychological Association, Washington, DC.

Card, S., Moran, T., & Newell, A. (1983). The psychology of human-computer interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

Card, S. K., Moran, T. P., & Newell, A. (1986). The model human processor. In K. R. Boff, L. Kaufman, & J. P. Thomas (Eds.), Handbook of Perception and Human Performance. Vol. 2. New York: Wiley.

Drucker, E. H., Cannon, L. D., & Ware, J. R. (1969). The effects of sleep deprivation on performance over a 48-hour period. HUMRRO Technical Report, 69-8.

Elliott, T. K. & Joyce, R. P. (1971). An experimental evaluation of a method for simplifying electronic maintenance. Human Factors, 13, 217-227.

Embrey, D. E. (1983). The use of performance shaping functions and quantified expert judgment in the evaluation of human reliability: An initial appraisal. Washington, DC: U. S. Nuclear Regulatory Commission.

Flexman, R. E. & Stark, E. A. (1987) Training simulators. In G. Salvendy (Ed.), Handbook of human factors, pp 1012-1038. New York: Wiley.

Foley, P. & Morey, N. (1987). Sensation, perception, and system design. In G. Salvendy (Ed.), Handbook of human factors. New York: Wiley.

Galton, F. et al. (1881). Report of the anthropometric committee. Report of the British Association for the Advancement of Science, 225-272.

Galton, F. et al. (1883). Final report of the anthropometric committee. Report of the British Association for the Advancement of Science, 253-306.

Glenn, F. (1975). Development of a Quantitative Display Reading Model: Model specifications (Analytics Technical Report 1117-G). Willow Grove, PA: Analytics, Inc.

Glenn, F. (1982). A discrete learning model for manual pursuit tracking. Proceedings of the IEEE 1982 -- International Conference on Cybernetics and Society.

Glenn, F., & Doane, S. (1982). Human operator simulator model of the NASA Terminal Configured Vehicle (TVC): Final report (Technical Report 1463.01A). Willow Grove, PA: Analytics, Inc.

Glenn, F., & Iavecchia, H. (1982a). Human Operator Simulator (HOS) programmer's guide (Technical Report 1400.22B). Willow Grove, PA: Analytics, Inc.

Glenn, F., & Iavecchia, H. (1982b). Human Operator Simulator (HOS) user's guide (Technical Report 1400.22C). Willow Grove, PA: Analytics, Inc.

Glenn, F., & Iavecchia, H. (1982c). Human Operator Simulator (HOS) study guide (Technical Report 1400.22E). Willow Grove, PA: Analytics, Inc.

Glenn, F., & Iavecchia, H. (1982d). Investigation of model/software enhancement to HOS (Technical Report 1400.22F). Willow Grove, PA: Analytics, Inc.

Giovani, B. & Goldman, R. F. (1971). Predicting metabolic energy cost. Journal of Applied Physiology, 30, 429.

Goldman, R. F. Prediction of human heat tolerance. In S. J. Folinsbee et al. (Eds.), Environmental stress. New York: Academic Press.

Haisman, M. F. & Goldman, R. F. (1974). Effect of terrain on the energy cost of walking with back loads and handcart loads. Journal of Applied Physiology, 36, 545-548.

Harmon, H. (1975). Modern Factor Analysis (3rd ed.). Chicago, IL: University of Chicago Press.

Harris, R., & Iavecchia, H. (1984). Crewstation Assessment of Reach revision IV (CAR-IV) user's guide (Technical Report 1800.10A), Willow Grove, PA: Analytics, Inc.

Harris, R., Glenn, F. A., Iavecchia, H., Rothenheber, E., & Zaklad, A. (1986). HOS-IV specifications (Technical report 1800.28B), Willow Grove, PA: Analytics, Inc..

Hockey, G. R. J. (1978). Effects of noise on human work efficiency. In D. N. May (Ed.), Handbook of noise assessment. New York: Van Nostrand.

Iavecchia, H., Harris, R., & Rothenheber, R. (1984). CAR-IV validation study results (Technical Report 1800.33B). Willow Grove, PA: Analytics, Inc.

Kobrick, J. L. & Tolcott, M. A. (1983). Climate and human performance. In D. J. Osborne & M . M. Gruneberg (Eds.), Psychology and productivity at work: The physical environment. London: Wiley.

Lane, N., Strieb, M., & Leyland, W. (1979). Modeling the human operator simulator: Applications to cost effectiveness. Modeling and Simulation of Avionics Systems and Command Control and Communications Systems. NATO-AGARD Conference Proceedings CP-268.

Laughery, R., Drews, C., Archer, R., & Kramme, K. (1986). A MicroSAINT simulation analyzing operator workload in a future attack helicopter. In National Aerospace and Electronics Conference. Dayton, OH: IEEE.

Levison, W. H. (1970). A model for task interference. In Proceedings of the 6th Annual Conference on Manual Control. Wright-Paterson AFB, OH.

Loeb, M., Holding, D. H., & Baker, M. A. (1982). Noise stress and circadian arousal in self-paced computation. Motivation Emotion, 6, 43-48.

Luria, A. (1966). Higher cortical functions in man. New York: Basic Books.

Meister, D. (1984). Human reliability. In F. Muckler (Ed.), Human Factors Review 1984. Santa Monica, CA: Human Factors Society.

Morris, N. M. & Rouse, W. B. (1985). Review and evaluation of empirical research in troubleshooting. Human Factors, 27(5), 503-530.

Navon, D. (1986). Visibility or disability: Notes on attention (ICS Report 8606). La Jolla, CA: University of Califomia at San Diego, Institute for Cognitive Science.

Opstad, P. K., Ekanger, R., Numme-Stad, M. & Raabe, N. (1978). Performance, mood, and clinical symptoms in men exposed to prolonged, severe physical work and sleep deprivation. Aviation Space Environmental Medicine, 49, 1065-1073.

Pew, R. (1969). The speed-accuracy operating characteristic. Acta Psychologica, 30, 16-26.

Pleban, R. J., Thomas, D. A., & Thompson, H. L. (1985). Physical fitness as a moderator of cognitive work capacity and fatigue onset under sustained combat-like operations. Behavior Research Methods, Instruments, and Computers, 17, 86-89.

Ramsey, J. D. (1983). Heat and cold. In G. R. J. Hockey (Ed.), Stress and fatigue in human performance. Chichester: Wiley.

Rohles, F. H. & Konz, S. A. (1987). Climate. In G. Salvendy (Ed.), Handbook of human factors, pp. 696-707. New York: Wiley.

Rouse, W. B. (1980). Systems Engineering Models of Human-Machine Interaction. New York: Elsevier.

Sage, A. (1981). Behaviroal and organization considerations in the design of information systems and processes for planning and decision support. IEEE Transactions on Systems, Man and Cybernetics, SM-11, 640-678.

Senders, J. W., Elkind, J. I., Grignetti, M. C. & Smallwood, R. (1966). An investigation of the visual sampling behavior of human observers. NASA CR-434.

Spady, A. A., Jr. (1978). Airline pilot scan patterns during simulated ILS approaches. (TP-1250). Washington, DC: NASA.

Strieb, M., Glenn, F., & Fisher, C. (1976). Human Operator Simulator: Volume VII - LAMPS Air Tactical Officer simulation (Technical Report 1200). Willow Grove, PA: Analytics, Inc.

Toquam, J., Dunnette, M., Corpe, V., McHenry, J., Keyes, M., McGue, M., Houston, J., Russell, T., & Hansen, M. *(1985). Development of cognitive/perceptual measures:* Supplementing the ASVAB. Paper presented at Annual Meeting of the American Psychological Association, Los Angeles, CA.

Wherry, R. Jr. (1986). Internal processes required for using displayed tactical information: Study number 1 (Technical Report 1800.31-TR-06). Willow Grove, PA: Analytics, Inc.

White, R., & Churchill, E. (1966). The body size of soldiers: U.S. Army anthropometry-1966 (Technical Report TR 72-51-CE). Natick, MA: U.S. Army Natick Laboratories.

Wohl, J. G. (1982). Maintainability prediction revisited: Diagnostic behavior, system complexity, and repair time. IEEE transactions on Systems, Man and Cybernetics, SMC-12, pp. 241-250.

Wylie, C. D., Mackie, R. R., & Smith, M. I. (1985). Comparative effects of 19 stressors on task performance: Major results of the operator survey. Proceedings of the Human Factors Society 29th Annual Meeting. Santa Monica, CA: Human Factors Society.

APPENDIX A

Some System Specifications

Coordinated by Jonathan Kaplan, Ph.D.
Army Research Institute

SYSTEMS INTEGRATION GUIDELINES III

All guidelines refer to the products being developed as part of the ARI MANPRINT Methods Development Program.

A-1. Hardware

All software to be developed should be able to run on the same hardware. The hardware of choice has the following characteristics:

a. Enhanced graphics display.

b. Enhanced graphics board 256 KB RAM.

c. 80286 processor.

d. Hard disk of with a minimum of 20 megabytes of storage.

e. Up to four megabytes of enhanced memory. Memory to conform to EMS standard.

f. Bernoulli Box or its functional equivalent with two removable 20 meg disks.

g. 80287 coprocessor chip for intensive floating point computations.

h. 1200/2400 baud internal modem that conforms to Hayes standards.

i. Floppy drive(s) capable of read/write to 360 kilobyte floppy diskettes.

j. Dot matrix printer with 132 characters per line capability. Printer must be capable of emulating IBM Graphics and Epson FX and LQ series printers.

k. Mouse and mouse drivers as required. Drivers for major brands of mice (in bus and com port configurations) should be provided.

l. An AT keyboard.

A-2. Operating Systems

All software to be developed should run under the same operating system. Until further developments result in an available, multitasking operating system, the DOS of choice will be DOS 3.2. Any requirement for contacting more than 640 kilobytes of memory will be handled via the EMS standard for enhanced memory.

A-3. Product Interactions

In many cases, the output of one of the products will be used as input by another. This means that each product must be able to convert its output 'ɔ a form (like ASCII) that can be understood by any other product that needs it, and some mechanism for moving such files from one machine to another must be provided. All products must have alternative mechanisms for developing and/or inputting these data. Potential product interactions are as follows:

a. Product 1 receives no input from any of the other products.

b. Product 2 receives missions from product 1.

c. Micro Analysis & Design-DRC's product 3 receives MOS, and maximum manpower requirements across systems from product 2.

d. Perceptronics-AIR's product 3 receives missions, functions, tasks, conditions, and criteria from product 1. It receives MOS from product 2.

e. Product 4 receives functions, tasks, and criteria from product 1.

f. Micro Analysis & Designs-DRC's product 5 receives tasks from product 1 (as a fall back).

g. ASA-SAIC's product 5 receives tasks from product 1 (as a fall back).

h. Micro Analysis & Design-DRC's product 6 receives missions, functions, tasks, jobs, number of individuals per system, task performance times, performance criteria, and task sequences from product 5. It receives training times and gross types of training media per function/task from product 4. It receives a measure of central tendency of each soldier characteristic from product 3.

i. Analytics' product 6 receives missions, functions, tasks, conditions, and criteria from product 1. Training times and media per function/task are received from product 4. Task times and jobs (described as groups of tasks) are received from product 5.

j. Perceptronics-Hay's product 6 receives missions, functions, tasks, conditions and criteria from product 1. It receives jobs (described as groups of tasks) from product 5.

A-4. Software

It is acceptable to use off the shelf software packages as components of the various products. Any such software must be available to the government through existing contracts. It is preferable to keep the number of software packages that the government must purchase to a minimum. At present, it is assumed that individual users of the various products will purchase their own off the shelf software packages (rather than via an ARI licensing agreement). However, the software produced for the six products will, without alteration, be able to communicate with all required off the shelf software.

Software and data bases will be available on Bemoulli discs. However, users will be offered the option (by software) to read from and write to the system's hard disk, if they choose not to purchase a Bemoulli. Master Bernoulli disks will be kept unused and will be copied for use. To allow relatively easy alterations of product output thus permitting sensitivity analyses and other "what if" games, intermediate files will be saved. All such intermediate and final output files will be saved to Bemoulli disk for security purposes.

If social security numbers are used in data bases that are intemal to any of these products, they will be encrypted. This encryption will permit linkage to external data sources, but will prevent product users from identifying the specific individuals to whom such data is linked.

Two commercial database management systems have been selected: R-Base V, and Pick/Revelation. No additional DBMS can be included as a product if this inclusion would result in product users having to purchase a third DBMS. Writing a DBMS to serve as a component of a product is acceptable.

In addition, such a database must be capable of editing inputs from other products and must be accessible via calls from Microsoft C. All users of DBMS components in their products must provide all other teams with the software required to: open, access, and close their DBMS data/files. This software must be developed and provided no later than six months following the start of Phase III. All products will output files in delimited fixed ASCII according to Data Interchange Format (DIF) to permit data communications.

The development language of choice is Microsoft C.

A **"Data Dictionary"** will be developed by the various product design teams to facilitate data communications among the products. This dictionary will be completed in three months from the start of Task 2. This dictionary will have the following classes of information.

a. Field names.

b. Length per field.

c. Type per field (alpha, numeric, etc.).

d. Comment per field (description of contents).

e. "Parent-child" relationship per field (Where does it come from and where does it go in a hierarchy).

f. Range per field.

g. Name of each record.

h. Estimated numbers of records.

A-5. User Interfaces

It is critical that the user interface(s) of all the MANPRINT products be designed in a manner that makes their operation as simple and self-evident as possible. The object of these products is to aid personnel who have limited or no computer experience. Six of the ten design teams have specified their intention to develop a common interface based upon the interface of Borland's Turbo C. The following requirements apply to any interface that is to be used by any product. They are mandatory.

a. The user will not have to memorize command language.

b. Training will be handled by a "self-evident" interface and/or built in help. Whenever external training (documentation or instruction) is called for, it must be accompanied by an explanation as to why this could not be accomplished by the interface and/or help.

c. If a mouse is used, equivalent keyboard entries must be available to the user.

d. Vocabulary meaning must be consistent across all products. That is, whenever a term (condition, criterion, characteristic, etc.) is used, it must always mean the same thing in each product. An on line glossary providing definitions of all key terms will be available as part of each product.

e. If an interface requires use of hierarchically nested menus that are more than two menus deep, a mechanism must be provided to the user to show him where in the menu structure he is at any time. This mechanism must be common across all products (if used).

f. If a given product requires more total effort of its user than can be done in a continuous three hour period, then the interface must provide the following two features:

(1) A mechanism for returning to the last procedure after the system has been turned off.

(2) A mechanism for seeing which procedures have been done and which have not been done at any given time.

g. To the extent possible, natural language will be used. That is, neither computer nor psychology jargon will be used to communicate unless a given word is now in the common domain.

h. If color coding is used, it must have the same meaning across all products.

i. If function keys are used, they must have the same effects across all products if at all possible. If such communality is impossible, another sort of communication mechanism should be considered.

j. Housekeeping procedures (starting, closing, saving, restoring, etc.) should be identical across products from the point of view of the user.

k. Users should be given the option of changing the foreground vs background colors. That is, dark letters-light background or the reverse.

l. Each product will be used more than once. Therefore, the names of the files that are generated must be able to be viewed and selected. The procedures for doing this should be as similar as possible across products (from the point of view of the user).

m. Each product must include an enhanced graphics driver, mouse drivers and printer drivers that will operate at minimum IBM, and Epson FX and LQ series printers.

n. From the point of view of the user, all editing conventions should be the same across products. Editing includes: entering, deleting, altering, moving, and copying text. Editing conventions include the selection of keys for moving the cursor, deleting, entering, copying, etc. Editing conventions should be as simple and self-evident as possible.

APPENDIX B

USER COMPUTER INTERFACE TECHNOLOGIES

B. User Computer Interface Technologies

The SEPTA system will provide many powerful functions to the user to assist them in their role as system analysts. However, these functions will go to waste if the user finds it difficult to communicate with SEPTA, specify the characteristics of their problems, or decipher the results. Therefore, the nature of the user-interface is a critical determinant to the successful use of SEPTA. This section presents the results of research conducted into several critical areas for the user interface, namely the user-computer dialog and user profiling.

This discussion of user interfaces assumes that SEPTA will be developed on a PC-AT class of microcomputers and that the microcomputer is equipped with 1) an enhanced color graphics monitor with an enhanced color graphics adapter and 2) a pointing device such as a mouse. A mouse is a small box (about the size of a deck of playing cards) with one to three buttons on its top face that functions as a pointing device. The mouse motion translates into corresponding movements of a pointer (e.g., a cursor) on a display. It allows the user to manipulate information and select commands or locations on the screen without having to enter specialized interface commands via the keyboard.

Since it was decided that SEPTA is to provide a highly visual interface, particular attention was paid to guidelines available for designing interactive graphic interfaces. Fairly detailed and comprehensive guidelines have been developed for various areas of UCI's, including keyboard design, display legibility, and ergonomic considerations, that are based on empirical data which has been organized in several reference handbooks (Engle, 1975, and Brown, 1983). Unfortunately, few guidelines are available to assist the designer in developing interactive interfaces that take full advantage of the features accorded by a highly graphic system. This section will present "guiding principles" but it is expected that the exact nature of the user interface will evolve iteratively in conjunction with the development of the SEPTA prototype through early and extensive involvement of the proposed user community and their feedback. This interaction between the SEPTA designers and the system analyst community is critical to the successful development of the system in order to develop the SEPTA UCI so that is effectively meets the needs of the user. The exact nature of the user interface will also be considered in conjunction with system requirements such as time criteria. Sophisticated user interfaces frequently make heavy demands on the system and and trade-offs may be required to ensure that the benefits of the user interface are not nullified by slow system response time.

This section presents a compendium of the following interface design issues considered relevant for the SEPTA UCI and any available design principles:

- **Interactive Dialogue** — identify the techniques used for communications between the user and SEPTA including dialogue style and user aids.
- **Input Capabilities** — identify the apprcpriate input devices (e.g., keyboard, mouse, etc.) and their use for user communication with the system for the system analyst functions.
- **Output Capabilities** — determine how information is to be presented to the user in such a way that maximizes the user's perception and understanding of information including screen design and layout, use of color, and use of graphics.

B.1 Intelligent User Dialogue

One of the key issues in designing a user interface is the selection of the techniques used for communications between the user and the computer. This not only involves selection of a dialogue type, but the development of techniques to assist and guide the user's interactions with the computer system. As described in Section 2, SEPTA users will possess varying levels of computer usage skills and system analyst expertise. The user-computer interface (UCI) must supply a meaningful structure within which the two-way dialogue between the user and SEPTA occurs.

B.1.1 Dialogue Types

The user communicates with the system by specifying commands and objects to be manipulated. There are six primary dialogue types:

1. Command-Driven,
2. Menu-Driven,
3. Question-Answer,
4. Form Fill-in,
5. Graphics-Driven, and
6. Natural Language.

Each dialogue structure type has a particular user appeal depending on a user's knowledge of the system and computer expertise. Many systems are now based on a hybrid of these techniques.

Command-Driven Dialogues. Command-driven dialogues typically display a brief prompt to the user (e.g., a question mark) and expect the user to type in a command name, phrase, or associated mnemonic (e.g., PRINT or PR). Since the system offers very few prompts or choices, the user is expected to know which system features are currently available and their syntax. If the system does not recognize a command that the user enters, it typically responds with an error message. The biggest advantage of a command-driven interface is speed. Very few keystrokes are required to initiate any action, and it eliminates stepping through multip'e levels of menus to specify an action. Command-driven interfaces appeal to experienced users since the system functions are more readily accessible. Command dialogue's biggest shortcoming is its inherent "unfriendliness." A system or computer novice viewing a display with nothing but a prompt has no guide as to what to do next. Novice users have many problems learning a command dialogue system since they are unfamiliar with both the functions that can be accomplished and the command names required to invoke the functions. Additionally, in order to invoke a command, the user has to first remember the designated command. If there are too many commands to be able to remember them easily, users tend to find this facility too frustrating and time-consuming to use.

Menu-Driven Interfaces. Menu-driven interfaces display every possible choice that the user can make in a menu that is typically arranged in a hierarchical manner (i.e., one choice or action must be taken before another). The selection of one menu item generates another menu, which brings up yet another until a final selection allows the desired function to be accomplished. The major advantage provided by menu dialogues is the ability to guide a user through the steps needed to accomplish a task. Menus reduce memory demands because they only require the user to recognize rather than recall the correct option (Martin, 1973). However, menu-driven interfaces are not without problems. New users might find that learning larger systems is difficult because information must be integrated across a series of displays. As each menu is viewed in isolation, relationships between menus are difficult to grasp and the user may get lost in the hierarchical structure. However, menu interfaces do facilitate use by novices. On the other hand, experienced users are often frustrated when they must step through a number of menus before reaching the desired function because not all functions are available from all menus.

A strategy that is effective for workstations with display windowing and mouse capabilities is the pull-down menu that combines command and menu-driven techniques. Pull-down menus display a command bar at the top of a window with a submenu displayed in a new window box beneath it showing all available options. The user can use the mouse to move a pointer to one of the choices displayed in the box. As the pointer moves among the menu options, the current selection is highlighted and the user depresses a keyboard or mouse key to indicate the desired selection.

Question and Answer Dialogues. Question and Answer (Q&A) dialogues present the user with a series of questions to which the user responds one at a time. The question and answer process is repeated until the system has received the necessary information. Q&A dialogues typically decide the next question based upon the answer(s) to the previous question(s). Some incorporate a degree of natural language capabilities in order to avoid simple yes/no responses. If the system cannot understand a response or require additional information, clarification questions may be generated. Similarly, if the user cannot understand a question, an explanation facility is typically provided. Q&A dialogues are typically used in advice providing expert systems.

Q&A dialogues are most successful with novices who are unfamiliar with the problem to be solved. However, experienced users quickly become impatient when forced to step through a lot of questions. One way to alleviate this problem is to provide multiple modes of use, e.g., full sentence mode and abbreviation mode. Additionally, default response can be set for the particular user (this is described in more detail in Section B.2) as part of their "user profile." An additional consideration is how to permit the user to change the response to a previous question. Finally, the ability of Q&A dialogues to be utilized within the time constraints imposed on the system analysts needs to be considered since Q&A dialogue techniques require a certain amount of time per question. This fact generally precludes their usage for time critical applications.

Form Fill-In Interfaces. Form fill-in (or fill-in-the blank) interfaces provide the user with input forms in which the user enters necessary commands and data. A display of labeled fields and an area for entry of input are shown and the user moves the cursor between the input areas and enters the appropriate information. This type of design is well suited when there is a correspondence between the input display and paper forms familiar to the user. This type of interface requires the user to be cognizant of the field labels, permissible field values, and the data entry techniques to be employed in moving among the displayed fields. Form fill-ins are most appropriate for frequent users.

A variant of form fill-ins are input templates developed on computers with graphics capabilities. In addition to boxes for entry of text material, various fields can be defined to be check boxes for on/off parameters, click boxes for parameters containing ranges of numbers, and radio buttons for the user to select only one of a set of parameters. Recent research in the area of input templates, such as those used on the Apple Macintosh computer, will be considered in the development of the user interface (Smith et al., 1982; Norman & Draper, 1986)

Graphics-Driven Interfaces. Graphics-driven interfaces are a fairly recent development and generally provide an interface that is both "friendly" and nonrestrictive. Graphic icons replace or supplement words as command designators in menu-based systems. Instead of a temporal order display such as a menu, the user is presented with a spatial order of possible actions that are represented by "iconic" or pictorial representations of actions. The user positions a graphics pointer, such as a mouse, over the icon representing the desired action. Icons take advantage of the human ability to discern pictorial differences more quickly and easily than textual differences. However, the icons must be designed carefully in order to maximize their usefulness and are best suited when a limited set of clearly distinguishable options are available. The visual interfaces made possible by iconic representation provide an object orientation rather than a procedure orientation and enhance the user's knowledge about the system and its capabilities. However, there currently exists a very limited body of information about the effectiveness of iconic representations and how they should be designed to maximize information transfer.

Natural Language Interfaces. Natural language interfaces provide the ability of the computer software to process plain English user requests. English is a very complex language and contains many structural and semantic ambiguities. This complexity requires a vast amount of knowledge in order to understand even a simple sentence. Although unrestricted natural language is the least limiting to the user, it is not currently practical to timely machine-parse with available computer systems and technology because of it s extreme complexity. Many advances are being made in the field of natural language processing but it is doubtful that they could be incorporated into SEPTA in the near future without significantly affecting the machine resources available to the primary function of SEPTA, i.e., assisting the system analyst in decision-making functions. Additionally, a body of evidence suggests that limiting the vocabulary and syntax available to the user improves the user's ability to utilize and comprehend the system language (Bailey, 1985; Hendler & Michaelis, 1983). Other studies have shown that arbitrarily approximate treatments of natural language may cause more problems than they solve (Brown, Burton, & deKleer, 1982).

User Dialogue Recommendations. It is recommended that the initial dialogue style for the prototype SEPTA system be a combination of menu bars and input templates. Menu-bars can be used to show a list of available SEPTA functions on a line at the top of the screen. Pull-down menus can !ist submenus in a separate window displayed beneath the menu bar and will contain the list of commands available for a particular selection from the menu bar. Once the user has specified the SEPTA function, additional dialogue with the user will occur using a series of input templates. As described above, the exact nature of the UCI will be developed iteratively in conjunction with the user community.

B.2 User Aids

User aids provide the means to assist the user in making effective use of the system. They must be implemented in a consistent manner both for the user indicating that additional assistance is needed and in the form of the assistance.

B.2.1 HELP Keys

Supplementary on-line guidance in the form of brief command summaries and tool descriptions which assist the user in making decisions is an essential feature of a user-friendly system. HELP displays serve as cognitive development tools to assist the user's understanding of the system. A simple, standard action that is always available to the user should be developed to obtain HELP messages. For example, HELP might be requested by an appropriately labeled function key, by selection of an always available menu option, or by keying a question mark into a displayed entry area. The content of the HELP response should be related to the contents of the current display or the current sequence of activity. The explanations should be developed in conjunction with the user community in order to tailor the response to the user's likely needs.

B.2.2 Default Values

Default values represent the systems best guess of the responses expected by users. They must be developed carefully to represent clear, logical, and meaningful values. The use of default values can both speed data entry and reduce input errors for a defined task. The default values should be clearly displayed in a consistent manner. The user should have the capability to define, change, or remove default values easily for any data entry field. In addition to the default values specified by the system developers, the user should also be allowed to override and specify defaults were appropriate — such as the specification of the units of measurement.

B.2.3 Status Information

Status (or progress) information which signifies that the system is performing·a time-consuming function and which also provides an approximate indication of when the function will be completed is also an important user aid. Information is provided to the user to indicate that the system is actually doing something and is not waiting for any user actions in order to continue execution. Since the system analyst may be interrupted at any point to perform a more critical task, the availability of status information will allow the analyst to make a determination about whether to continue or terminate the current function.

B.3 Input Capabilities

B.3.1 Input Devices

Alphanumeric keyboards used for data entry should conform to the Qwerty arrangement and the keyboard should include a keypad to assist in entry of numeric data (Van Cott & Kinkade, 1972).

The UCI also requires the use of a device that can point quickly to items on the display and that are faster than the directional cursor keys such as a mouse, joystick, trackball, light pen, etc. Of these devices that are commonly available, the mouse has been identified as the preferred one for most video pointing needs. The mouse is a hand-held pointing device with a sensor on the bottom to detect motion over a flat surface and one to three buttons on the top which can be sensed by system software. The system provides the user with continuous feedback as to where it thinks the mouse is pointing by displaying a cursor on the screen. The user slides the mouse around on a flat surface (causing the bearings on the bottom of the mouse to rotate), and the system moves the cursor on the display. The user indicates that the mouse has positioned the cursor at the desired location by pressing a button on the top of the mouse.

The most common use of a mouse is to move a cursor to a precise screen location and depress a mouse button in order to initiate some action, such as selecting a menu option, that otherwise would require the depressions of special function keys and/or directional cursor movement keys. The mouse is the preferred pointing device for text-editing applications because it is the most comfortable to use and is also among the fastest (Card et al, 1978; Warfield, 1983).

B.3.2 Data Entry

Data entry concerns the ways the user should be able to communicate with the system when entering data. The key to effective data entry is to reduce the amount of information required to be entered by the user to the absolute minimum. The system can predict likely data values and the user can manipulate auxiliary input devices such as a mouse to point to the desired response instead of requiring the user to type the full response wherever possible. The development of standard templates for input of commonly occurring information greatly speeds data entry and reduces user errors. These templates should contain clearly indicated default values for the most commonly occurring cases. Data entry should also permit natural expression of values such as indicating numerical entry or, in a few cases, coordinate information in the form of X,Y,Z. Th cues or prompts for data entry should contain terminology that is used consistently throughout the entire system and the use of abbreviations should be avoided. When entering data base queries, the user should select the appropriate query verb and data base elements from a list displayed by the system and the query syntax should then be displayed so that the user understands clearly what information is required. For example, the pr rtions on the statement that must be supplied by the user should be displayed in a format ano color that contrasts with the SEPTA supplied identifiers.

Whenever the user completes an input, the system should provide immediate feedback and not leave the user wondering whether or not his entry has been received. If the input requires extended processing that will not be commensurate with the user's expectations of system response, an indicator should be displayed to communicate to the user that system processing is occurring.

Users will make errors and system software should deal appropriately with incorrect entries. Not only should the, incorrect entry be clearly indicated but the user should be provided with explicit guidance on what actions are required and how to abort the current process. The human factors guidelines indicate that the user should be provided with immediate feedback, directional guidance, and that informative messages should be displayed that pinpoint as close as possible the particular user entry that caused the error (Morland, 1983). An additional consideration is to require confirmation from the user before performing a potentially disastrous operation such as deleting a file to occur. The user should be able to cancel the operation without loss of data and resume the current processing function.

B.4 Output Capabilities

To fulfill a goal for effective presentation of information to the user, the design of the UCI output capabilities and features must minimize the effort required by the user to scan, perceive, and interpret the myriad data involved in a complex arena like system analysis.

B.4.1 Screen Layout

Screen format and content concern what information is to be placed on the screen, how it is presented, and the physical layout of information appearing on the display. A well-designed screen reflects the needs of its user community and the tasks to be accomplished via use of the system. A well-designed interface allows the user to place everything relevant for accomplishing a particular task on the screen. This section focuses on human considerations in screen design which are oriented towards **simplicity**, **clarity**, and **understandability**. Important human characteristics to be considered in screen design are:

- **Perception**: The awareness and understanding of the elements on the screen which help establish order and meaning. Eye fixation studies indicate that initially one's eyes usually move to the upper left corner of the display and then quickly move in a clockwise direction. During and following this movement, users are influenced by the symmetrical balance and weight of the titles, graphics, and text of the display. A cluttered or unclear screen creates the requirement that some effort must be expended in learning and understanding what is presented.
- **Memory**: Short-term memory is highly susceptible to interference and its contents are quickly replaced by new information. Its capacity is about seven items plus or minus two (Miller, 1956). An important memory consideration, with significant implications for screen design, is the limited ability to recall the significance of many colors and/or symbols.
- **Learning**: A design developed with the intent of minimizing human learning time can accelerate human performance. Learning can be enhanced if it provides complete and prompt feedback.
- **Individual Differences**: The design must permit people with widely varying skill levels to satisfactorily and easily learn to use the system.

One of the biggest problems of screen display design arises when too much information is put on the screen which leads to user confusion and increased error rates. Screens should provide only relevant information because the more information, the greater the competition among screen components for a user's attention. Visual search times will be longer and meaningful patterns more difficult to perceive if the screen floods a person with too much information. Therefore, only information relevant to the user's current need should be displayed

(Brown, 1980). The user should be able to control what information is on the screen through such features as scrolling and also be able to eliminate information that is no longer of interest (Brown, 1980; Martin, 1973).

The layout of the screen should be structured so that the amount of user confusion is reduced. The user's perception of the structure among the different areas and/or objects on the screen can be enhanced by the careful and consistent use of a variety of techniques including different intensity levels, colors, numbers, and letters. Specific areas of the screen should be designated for certain kinds of information, such as menu options, input fields, status information, etc., and these areas should be maintained consistently on all screens.

Windows. One of the most popular techniques to manage the information overflow problems of screen displays is the use of windows. Windows permit the simultaneous display of two or more sets of information on a single display screen. Each window is usually (but not always) delimited by a border which separates it from the rest of the data on the screen. The major benefit of the capability to display multiple windows on a display screen is the reduction of the load placed on the limited cognitive resources of users, particularly short-term memory. Reducing this load would free these cognitive resources for other tasks. The following kinds of tasks derive the most benefit from a windowing capability:

- The display of supplemental information relevant to the user's primary task, e.g., help messages. Typically, Help messages result in a totally new display screen overwriting the primary screen, thereby forcing the user to remember the situation requiring assistance (which is no longer displayed). Similarly, the user must remember the help message since once the user returns to the primary task, the help message is done. Alternatively, Help can be displayed in a separate window which can be viewed simultaneously with the primary screen.
- Monitoring changes. For example, the user can modify data in one window and then have the result immediately reflected in a graph displayed in another window.

Display Features. Contrasting display features (e.g., different intensities and character sizes, blinking, reverse images, color, etc.) can be used to draw attention to different screen components, items being processed, and urgent items. However, these features should be used in moderation since their overuse is very distracting to the user. Blinking is excellent for attention-attracting. However, it reduces legibility and is distracting. Its use should be limited to situations where a user must respond quickly and it should be turned off as soon as the user has responded (Smith & Goodwin, 1971). It is also recommended that instead of blinking a message (which will make the message difficult to read), an adjacent symbol should be blinked instead.

This would assist the user in remembering which parameters were changed. Inverse video is good for attention-getting but it also reduces legibility. It should be used with discretion mainly to call attention to errors or important screen components. Its use to highlight an individual item as it is selected by the user, such as indicating a menu selection, provides timely and accurate feedback to the user (Engel & Granda, 1975). Inverse video is recommended for error messages, classification markings, indication of special function keys, and menu option prompts.

Typeface. For display of text, the use of both upper and lower case letters assist the user's perception. Lower case fonts with initial letter in upper cases should be used wherever possible because this format provides greater legibility (Marcus, 1984). Several studies have found that regular-type text is read significantly faster than text in all capitals because words are perceived by the shape of their outline and not deciphered letter by letter (Brown et al., 1980). For brief captions, labels, and prompts, all upper case can be used for emphasis but should be used spartanly because their use results in a decrease in reading speed by as much as 13 percent (Marcus, 1982). The font size of the characters should be at minimum 7 by 9 pixels (Vartabedian, 1971) and should use a single, simple typeface such as Roman or Helvetica.

B.4.2 Use Of Color

The use of color involves the methods used for making consistent and effective use of color for differentiating screen components and as an attention-getter. The use of color can assist the user in understanding the logical structure of the data on the screen. As a visual code, it can assist in giving meaning to the data or information displayed, the differentiation between required and optional data, and the indication of incorrect responses. However, color must be used carefully because its use alone will not guarantee improved performance and its misuse may impair performance by distracting the user and interfering with their handling of information. Since the use of color is attention-getting, it may prove distracting to the user who may notice differences in color, regardless of whether the difference has any meaning. Users also tend to visually group items of the same color in a way that may conflict with the intention of color coding.

The human eye cannot effectively distinguish more than eight colors at one time. In general, the use of color in displays should be limited to between five and seven colors (Murch, 1981). This is based upon the studies done by cognitive scientists demonstrating that humans experience great difficulty in maintaining more than 5-7 elements simultaneously. As the number of colors (in a display) increases, the probability of confusion among colors also increases. The cognitive aspects of remembering the association between a particular color and what it stands for also increases the user's workload and, in turn, increases the time to respond to a specific color. Color meanings should also be consistent with the user's expectations (e.g., red means danger). Another rule of thumb for assigning colors to display features is to avoid the pairing of opponent colors such as red/green, yellow/blue, and black/white. These combinations may produce afterimages that degrade the legibility of other items on the display (Durrett & Trezone, 1982).

Another factor affecting color discrimination is the environment in which the system is to be used. The presence of artificial or natural lighting has an effect on foreground-to-background contrast. Additionally, as illumination decreases the visibility of certain colors also decreases. Therefore the selection of colors should be based on the luminance levels at the workstation.

In conclusion, color should be used mainly to differentiate screen components and fulfill an attention-getting role. If each color is to communicate a specific significance, then the number of colors should be limited to about 6 clearly discriminable colors. Most importantly, the use of color must be consistent in all screen displays. Recommendations for the use of color for the UCI are ordered so that bright colors emphasize important data and colors lacking brightness are used for less important data.

B.4.3 Graphics

Interactive, high-resolution color graphics are a feasible and cost-effective presentation medium for SEPTA because of the advancement of computer technology that permit the representation of data in a graphic form that can readily be stored, manipulated, and displayed by computers. Advanced graphics facilities can considerably enhance the communication between the user and the system by providing constant visual feedback to provide guidance to the user and improve the overall quality of the system. Graphic displays not only provide a technique for presenting data but also provide a mechanism by which the complexity of the information can be reduced by making interesting data points, trends, and relationships more apparent. Several research studies have indicated that:

- Spatial and visual information is easier to remember than verbal and textual information (White, 1983).
- Graphical presentations of problem data and solutions result in faster perception of changes from previous results and faster determination of problem solutions (Ives, 1982).

- Graphics support the rapid scanning of the implications of a course of action and permit rapid identification of the need for further exploration.

SEPTA will involve the both the dynamic and static display of graphic data. A major problem for the display of information is the large amounts of potentially relevant data that must be examined in order to find the significant information for the user's current problem analysis. Moreover, what is relevant depends on what the system analyst is currently doing which could change over time, and what the user is currently interested in. Graphic representations can assist this extraction process by making more apparent interesting data points, trends, and relationships. The most common method of presenting information graphically uses position and reference shapes (e.g., the x and y axes of a graph) to indicate the value of the data and codes (e.g., color, label, textures) to identify the data. Research indicates that color is a better data identifier than size, angle, or shape; texture codes are better suited for distinguishing the display of several data sets on a combined graph (e.g., dotted/dashed lines); and surface texture is useful for dividing the bars of a bar graph into their component parts (Morse, 1979).

However, graphics must be used carefully and determining the right mix of text and graphics is another area where few guidelines are available for the interface designer. The available guidelines indicate that a consistent graphic vocabulary is necessary and that legibility and readability require the careful use of color, symbols, and typefaces (Marcus, 1982).

B.5 User Profiles

Typically, each user of a computer system develops a personal methodology for interconnecting seemingly isolated techniques and strategies in using a specific system. Over time the user develops a great deal of problem-specific declarative and procedural knowledge and as a result the user becomes proficient in operating that particular system. During an individual's tenure in a specific position, he or she is exposed to various computer systems. The time and effort spent to learn a new system can be significant. The learning phase is not productive and can be considered "lost" time. If the time to learn could be reduced to a reasonably short amount of time, the lost time would be negligible and the user would be able to benefit more quickly from the system. Each user will also be applying the system to the needs of a particular organizational level. Since SEPTA will be applicable to all analysts, it is important for it to include a capability to communicate with the user using a unique terminology applicable to the user's needs. For these reasons, it is recommended that SEPTA include the capability to construct a user profile. The user profile module will be a tool designed to capture user preferences, store them, represent them, and permit the user to interact with SEPTA in a customized manner that accommodates that particular user's knowledge rather than being restricted by system defined commands and protocols.

This user profile capability would permit a computer user to tailor the SEPTA interface by allowing the user to specify the terms used for communicating with SEPTA using his/her particular jargon. This tailoring tool would also significantly reduce the training time required and therefore reduce the cost associated with training. For novice users, user profiles can assist in eliminating computer anxiety so that the individual will be willing to learn a new system rather than expend time and effort avoiding it. For the expert user, it will allow them to use their personal preferences in constructing their user protocol which will result in increased productivity.

Once the issues that need to be addressed initially to construct the user profile have been resolved, the potential techniques employed to cause the system to recognize a particular user and then load the user's profile into the system working memory will need to be explored. Issues concerning maintaining, updating, and reconfiguring a user profile, once it has been initialized, will require investigation with regard to the various functions of the operating systems, the issues of compatibility, usability, and portability.

There are a few techniques currently available that allow a user to define system commands through the use of personal preference, such as macro and command files. However, such files are awkward to construct and may not be obvious to the infrequent or inexperienced user. Since many users do not possess a programming background, a criterion level of understanding needs to be established and then the degree of user control can be investigated. In addition, other variables, such as hardware limitations, may affect the degree of user freedom possible, and therefore, factors such as these need to be examined.

BIBLIOGRAPHY

Alavi, M. (1984). An assessment of the prototyping approach to information systems development. Communications of the ACM, 27(6), 556-563.

Andrews, E.W. (1983). *Trackball-interfacing techniques for microprocessors.* Byte, 8(12), 234-242.

Atwood, M.E. (1984). A report on the Vail workshop on human factors in computer systems. IEEE Computer Graphics & Applications, 4(12), 48-66.

Bailey, G.D. (1985). The effects of restricted syntax on human-computer interaction. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics,* 24-28.

Barto, A. and Anandan, P. (1985). Pattern-recognizing stochastic learning automata. IEEE Transactions on Systems, Man, and Cybernetics, 15(3), pp. 360-375.

Bass, L.J. (1985). A generalized user interface for applications programs (II). , Communications of the ACM, 28(6), 617-627.

Benbasat, I., and Wand, Y. (1984) Command abbreviation behavior in human-computer interaction. Communications of the ACM, 27(4), 376-383.

Bennett, J. (1984). Managing to Meet Usability Requirements: Establishing and Meeting Software Development Goals. In J. Bennett, D. Case, J. Sandelin, and M. Smith, (editors). Visual Display Terminals. Englewood Cliffs, NJ: Prentice-Hall, pp. 161-184.

Billingsley, P.A. (1982) *Navigation through hierarchical menu structures: Does it help to have a map?* Proceedings of the Human Factors Society , 103-107.

Biswas, G., Oliff, M. and Sen, A. (1985). Design of an expert system in operations analysis. IEEE Computer, pp. 121-125.

Brachman, R. (1983). What IS-A is and isn't: An Analysis of Taxonomic Links in Semantic Networks. IEEE Computer, 16(10), pp. 30-36.

Brachman, R., (1985). "I Lied about the Trees" or, Defaults and Definitions in Knowledge Representation. The AI Magazine. Fall, pp 80 - 92.

Brachman, R., and Levessque, H. (Eds.). (1985). Readings in Knowledge Representation. Los Altos, CA: Morgan and Kaufmann Publishers.

Brown, C.M., et al. (1983). Human Factors Engineering Standards for Information Processing Systems . Lockheed Technical Report LMSC-D877141. Sunnyvale, CA, Lockheed Missiles and Space Company.

Brown, J.W. (1982). Controlling the complexity of menu networks. Communications of the ACM, 25(7), 412-418.

Brown, P.J. (1983). Error messages: The neglected area of the man/machine interface. Communications of the ACM, 26(4), 246-249.

Buchana, B., and Shortliffe, E. (1985). Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project. Reading, MA: Addison-Wesley.

Carbonell, J. (1983). Learning by Analogy: Formulating and Generalizing Plans from Past Experience. Machine Learning: An Artificial Intelligence Approach. Palo Alto, CA: Tioga Publishing Company, pp. 137-161.

Campbell, A. J., et al. (1981). Reading speed and test production: A note on right-justification techniques. Ergonomics , 24(8), 633-640.

Card, S.K., et al. (1983). The Psychology of Human-Computer Interaction. Hillsdale, NJ: Erlbaum.

Card, S.K., et al. (1978). Evaluation of mouse, data-controlled isometric joystick, step keys, and text keys for text selection on a CRT, Ergonomics , 21(8), 601-613.

Carroll, J.M. and Carrithers, C. (1984). Training wheels in a user interface. Communications of the ACM, 27(8), 800-806.

Charniak, Eugene, Drew McDermott (1985). Introduction to Artificial Intelligence. Reading, MA: Addison-Wesley Publishing Company, p. 460.

Chignell, M.H., et al. (1985). Intelligent interface design. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 620-623.

Chouraqui, E. (1985). Construction of a model for reasoning by analogy. Progress in Artificial Intelligence, Steels, L. and Campbell, A. (Eds.). Chichester, England: Ellis Horwood Limited .

Clocksin, W. and Mellish, C. (1984). Programming in Prolog, 2nd Edition. Springer-Verlag.

Cohen, P. and Feigenbaum, E. (1982). The Handbook of Artificial Intelligence, Volume III. Stanford, CA: HeurisTech Press .

Copi, I. (1954). Symbolic Logic. New York, NY: The MacMillan Company.

Davis, S.E., et al. (1985). An experimental comparison of a windowed vs. a nonwindowed operating system environment. Proceedings of the Human Factors Society, 250-254.

Dumais, S.T., and Landauer, T.K. (1983). Using examples to describe categories. Chi-83. Proceedings of the Conference on Human Factors in Computer Systems, 112-115.

Durham, I., et al. (1983). Spelling correction in user interfaces. Communications of the ACM, 26(10), 764-773.

Durrett, J., and Trezona, J. (1982). How to use color displays effectively. Byte, 7(4), 50-53.

Eberts, R. (1985). Four approaches to human-computer interaction. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 615-619.

Eliot, L. (1986). Analogical Problem-Solving and Expert Systems. IEEE Expert, Summer, pp. 17-26.

Engle, S.E. and Granda, R.E. (1975). Guidelines for man/display interfaces, Technical Report TR00.2720, Poughkeepsie, NY: IBM.

Evans, T. (1968). A heuristic program to solve geometric analogy problems. Semantic Information Processing, Minsky, M. (Ed.) Cambridge, MA: MIT Press.

Findler, N., Lo, R. and Bhaskaran, P., Two theoretical issues concerning expert systems. IEEE Computer, pp. 236 - 240.

Finin T (1986). Part II: Understanding Frame Languages Implementing PFL. AI Expert, December, pp. 51-56.

Fischler, M and Firschein, O. (1986). Intelligence The Eye, the Brain, and the Computer Reading, MA: Addison-Wesley.

Fish, R.S., et al. (1985). Tool sharpening: Designing a human-computer interface. Proceedings of the Human Factors Society, 475-479.

Fisher, D.L., et al. (1985). Optimizing the set of highlighted options on video display terminal menus. Proceedings of the Human Factors Society, 650-654.

Foley, J.D., et al. (1984). The human factors of computer graphics interaction techniques. IEEE Computer Graphics & Applications, 4(11), 13-48.

Freiling, M., Alexander, J., Messick, S., Rehfuss, S. and Shulman, S. (1985). Steps towards automating expert system development. IEEE Computer, pp. 985-993.

Fox, M., Roth, S., Sathi, A., and Mattis, J. (1986). Callisto: An intelligent project management system. The AI Magazine, Winter, pp. 34-52.

Galitz, W.O. (1985). Handbook of Screen Format Design . Wellesley Hills, MA: QED Information Sciences.

Gilfoil, D. (1982). Warming up to computers: A study of cognitive and affective interaction over time. Proceedings of the Conference on Human Factors in Computer Systems, 245-250.

Good, M.D. (1982). Building a user derived interface. Communications of the ACM, 27(10), 1032-1043.

Gould. J.D. (1968). Visual factors in the design of computer-controlled CRT displays, Human Factors 10, 359-375.

Gould. J.D., and Lewis , C. (1983). Designing for usability--key principles and what designers think chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 50-53.

Greenberg. S. and Witten, I (1985). Adaptive personalised interfaces - A questions of viability. Behaviour and Information Technology, (4), 31-45.

Haack, S. (1978). Philosophy of Logics. Cambridge, England: Syndics of the Cambridge University Press.

Hartzband, D. and Maryanski, F. (1985). Enhancing knowledge representation in engineering databases. IEEE Computer, September, pp. 39 - 48.

Hendler, J.A., and Michaelis. P.R. (1983) The effects of limited grammar on interactive natural language. chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 190-192.

Hillier, F., and Lieberman, G. (1974). Operations Research, 2nd Edition. San Franciso, CA: Holden-Day, Inc.

Houghton, R. (1984). On-line help systems: A Conspectus. Communications of the ACM, 27(2), 126-133.

Innocent, P.R. (1982). Towards self-adaptive interface systems. International Journal of Man-Machine Studies, 16, 287-299.

Ihara, Jiro (1987). Extension of Conditional Probability and Measures of Belief and Disbelief in a Hypothesis Based on Uncertain Evidence. IEEE Transactions on Pattern Analysis and Machine Learning, 9(4), pp. 561- 568.

Jacob, R.J.K. (1983a). Executable specifications for a human-computer interface. chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 28-34.

Jacob, R.J.K. (1983b). Using formal specifications in the design of a human-computer interface. Communications of the ACM, 26(4), 259-264.

Kahane, H. (1973). Logic and Philosophy. Belmont, CA: Wadsworth Publishing Co.

Kahn, G., Nowlan, S., and McDermott. (1985). Strategies for Knowledge Acquisition. IEEE Transations on Pattern Analysis and Machine Intelligence, volume PAMI-7(5), pp. 511-522.

Kelly, M.J., and Chapanis, A. (1977). Limited vocabulary natural language dialogue. International Journal of Man-Machine Studies, 9, 479-501.

Klaus, B. and Horn, P. (1986). Robot Vision. Cambridge, MA: Wadsworth Publishing Co.

Kullback, J. (1986) An expert system for ELINT analysis. WESTEX-86: Proceedings of the IEEE Western Conference on Knowledge-Based Engineering and Expert Systems, Anaheim: CA., 38-41.

Kumara, S. and Kashyap. R., Knowledge representation in expert systems via entity-relationships and its applications. IEEE Computer, pp. 495- 500.

Landauer, T.K., et al. (1983). Natural command names and initial learning: A study of text editing terms. Communications of the ACM, 26(7), 495-503.

Lecot K., Parker D. (1986). Control over Inexact Reasoning. AI Expert, Premier, pp. 32- 43.

Lewis, J.W., Ph.D. (1983). An effective graphics user interface for rules and inference mechanisms. chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 139-143.

Lipschutz, S. (1965). Probability. New York: McGraw-Hill Book Company.

Loveland, D. W. (1984). Knowledge Acquisition and Evaluation within Expert Systems, Defence Technical Information Center Technical Report: ADP003035. Defense Logistics Agency, Alexandria, VA:DTIC.

Maguire, M. (1982). An evaluation of published recommendations on the design of man-computer dialogues. International Journal of Man-Machine Studies, 16, 237-261.

Malone, T., Grant, K., Turbak, F., Brobst, S., and Cohen, M. (1967). Intelligent Information-Sharing Systems. Communications of the ACM, 30(5), pp. 390- 402.

Mandi, A., and Chu, Y. (1985). Network Model-Guided Expert Knowledge Elicitation. IEEE Computer, pp. 10-13.

Marcus, A. (1984). Corporate identity for iconic interface design: The graphic design perspective IEEE Computer Graphics & Applications, 4(12), 24-32.

Marcus, A. (1982). Designing the face of an interface. IEEE Computer Graphics and Applications , 2(1), 23-29.

Maron, M., Curry, S., and Thompson, P. (1986). An inductive search system: Theory, design, and implementation. IEEE Transactions on Systems, Man, and Cybernetics, 16(1), pp. 21 - 28.

Martin, J. (1973). Design of Man-Computer Dialogues. Englewood Cliffs, NJ: Prentice-Hall, Inc.

Mason, R., and Carey, T. (1983). Prototyping interactive information systems. Communications of the ACM, 26(5), pp. 347-354.

McCoy, K.F. (1983). Correcting misconceptions: What to say when the user is mistaken. chi-83: Proceedings of the Conference on Human Factors in Computer Systems , 197-201.

Meads, J.A. (1985). Friendly or frivolous. Datamation, 31(7), 96-100.

Michalski, R. and Stepp, R. (1983). Automated Construction of Classifications: Conceptual Clustering Versus Numerical Taxonomy. IEEE Transactions on Pattern Analysis and Machine Intelligence, 5(4).

Moran, T.P. (1981). The command language grammar: A representation for the user interface of interactive computer systems. International Journal of *Man-Machine Studies*, 3-50.

Morland, D.V. Human factors guidelines for terminal interface design. Communications of the ACM, 26(7), 484-494.

Morse, A. (1979). Some principles for the effective display of data. Computer Graphics , 13(2), 94-101.

Murch, G. (1983). The effective use of color: Cognitive principles. Tekniques , 8:2. 25-31.

Murch G. (1984a). Physiological principles for the effective use of color. IEEE Computer Graphics & Applications, 4(12), 49-54.

Murch, G. (1984b). The effective use of color: Cognitive principles. Tekniques, 8(2), 25-31.

Murch, G. (1984c). The effective use of color: Perceptual principles. Tekniques, 8(1), 4-9.

Myers, B.A. (1984). The user interface for sapphire. IEEE Computer Graphics & Applications, 4(12), 13-23.

Nakamura, K., and Iwai ,S. (1982). Topological fuzzy sets as a quantitative description of analogical inference and its application to question-answering systems for information retrieval. IEEE Transactions on Systems, Man, and Cybernetics, 12(2), pp. 558-568.

Nakamura, K., Sage, A., and Iwai, S. (1983). An intelligent data-base interface using psychological similarity between data. IEEE Transactions on Systems, Man, and Cybernetics, 13(4), pp 193-204.

Nakatani, L.H., and Rohrlich, J.A. (1983). Soft machines: A philosophy of user-computer interface design. Chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 19-23.

Neal, A.S., and Emmons, W.H. (1982). Operator corrections during text entry with word processing systems. Proceedings of the Human Factors Society 26th Annual Meeting , 625-628.

Negoita, C. (1985). Expert Systems and Fuzzy Systems. Menlo Park, CA: The Benjamin/Cummings Publishing Company, Inc.

Norman, D.A. (1983a). Design rules based on analyses of human error. Communications of the ACM, 26(4), 254-258.

Norman, D.A. (1983b). Design principles for human-computer interfaces. Chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 1-10.

Norman, D., and Draper, S. (Eds.). (1986). User Centered System Design: New perspectives on Human-Computer Interaction. Hillsdale, NJ: Lawrence Erlbaum Associates.

Olsen, D.R., Jr., et al. (1984). A context for user interface management. IEEE Computer Graphics & Applications, 4(12), 33-42.

Oppenheimer, J. (1956). Analogy in science. American Psychologist, Vol. 11.

Otto, H. (1978). The Linguistic Basis of Logic Translation. Washington, D.C: University Press of America.

Pitrat, J. (1984). An intelligent system can and must use declarative knowledge efficiently. Artificial and Human Intelligence, A. Elithorn, and R. Banerji (Eds.). Elsevier Science Publisher, pp. 271 - 280.

Prerau, D. (1985). Selection of an appropriate domain for an expert system. The AI Magazine, Summer, pp. 26 - 36.

Rada, R. (1985). Gradualness facilitates knowledge refinement. IEEE Transactions on Pattern Analysis and Machine Intelligence, 7(5), pp. 523- 530.

Ramsey, H.R., and Atwood, M.E. (1980). Man-computer interface design guidance: State of the art. Proceedings of the Human Factors Society, 85-89.

Rasmussen, J. (1985). The role of hierarchical knowledge representation in decisionmaking and system management. IEEE Transactions on Systems, Man, and Cybernetics, 15(2), pp. 234-243.

Rauch, H. (1984). Probability Concepts For An Expert System Used For Data Fusion. The AI Magazine, Fall.

Reisner, P. (1981). Formal grammar and human factors design of an interactive graphics system. IEEE Transactions on Software Engineering, SE-7(2), 229-240.

Reitman, W. (1965). Cognition and thought. New York: Wiley.

Rine, D. (1985) Some applications of fuzzy logic to future and expert systems. IEEE Computer, pp. 351 - 356.

Roach, J.W. (1985). Adapting to individual users: The user-trainable interface. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 228-235.

Roach, J.W., and Nickson, M. (1983). Formal specifications for modeling and developing human/computer interfaces. Chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 35-39.

Rouse, W.B. (1981). Human-computer interaction in the control of dynamic systems. Computing Surveys, 13(1), 71-99.

Rumelhart, D., and Abrahamson, A. (1973). A model for analogical reasoning. Cognitive Psychology, Vol. 5.

Rushinek, A., and Rushinek, S. (1986). What makes users happy? Communications of the ACM, 29.

Schank R. (1984). Memory-Based Expert Systems. Defence Technical Information Center Technical Report: AFOSR-TR-84-0814. Defense Logistics Agency, Alexandria, VA:DTIC.

Schnirring, B. (1986). Artificial Intelligence. NASA Tech Briefs, Special Edition, pp. 24- 30.

Shen H. and Chan K. (1985). An Aid for Knowledge Acquisition in Knowledge-Based Systems, Pattern Analysis and Machine Intelligence Group. Department of Systems Design Engineering, University of Waterloo, Waterloo, Canada.

Shinar, D., et al. (1985). The relative effectiveness of alternative selection strategies in menu-driven computer programs. Proceedings of the Human Factors Society, 645-649.

Shneiderman, B. (1986). Design the User Interface: Strategies for Effective Human-Computer Interaction. Reading, MA: Addison-Wesley.

Shneiderman, B. (1982). Designing computer system messages. Communications of the ACM, 25(9), 610-611.

Shneiderman, B. (1980). Software Psychology: Human Factors in Computer and Information Systems. Cambridge, MA: Winthrop.

Shulman, H.G. (1985). Icons versus names as command designations in text editing. Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 268-272.

Silverman, B. (1983). Analogy in Systems Management: A Theoretical Inquiry. IEEE Transactions on Systems, Man, and Cybernetics, 13(6), pp. 1049-1075.

Silverman, B. (1985). The Use of Analogs in the Innovation Process: A Software Engineering Protocol Analysis. IEEE Transactions on Systems, Man, and Cybernetics, 15(1), pp. 30-44.

Smith, C., Irby, C., Kimball, R., Verplan, B., and Harslem, E. (1982). Design the Star User Interface. Byte, 7(4), pp. 242-282.

Smith, S.L. Man-machine interface requirements definition: Task demands and functional capabilities. *Proceedings of the Human Factor Society*, 93-97.

Smith, S.L. and Goodwin, N.C. (1971). Blinking coding for information displays, (Human Factors), 13, 283-290.

Smith, S.L. and Mosier, J.N. (1984). Design Guidelines for User-System Interface Software Bedford, MA: Mitre Corporation Technical Report MTR-9420.

Somberg, B.L., and Dotson, K.E. (1985). Methods of disambiguating command term abbreviations for interactive computer systems. Proceedings of the Human Factor Society, 659-663.

Spearman, C. (1927). The Abilities of Man, New York, NY: Macmillan.

Steinhaus, C.P., and Hammer, J.M. Development principles for the construction of help systems. *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics*, 375-379.

Sternberg, R. (1977). Component Processes in Analogical Reasoning. Psychological Review, Vol. 84, pp. 353-378.

Strawson, P.F. (1966). Introduction to Logical Theory, Methuen & Co LTD. London: 1966

Swezey, R.W. and David, E.G. (1983). A case study of human factors guidelines in computer graphics, IEEE Computer Graphics & Applications, 3(11), 21-30.

Teitelbaum, R.C., and Granda, R.E. (1983) The effects of positional constancy on searching menus for information. Chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 150-153.

Tennant H.R., et al. (1983). Usable natural language interface through menu-based natural language understanding. Chi-83: Proceedings of the Conference on Human Factors in Computer Systems, 154-165.

Trevellyan, R., and Browne, D. (1987). A self-regulating adaptive system. In Proceedings of the 1987 Human Factors in Computing Systems and Graphics Interface, pp 103-107.

US Army (1983). Military Intelligence Group (Combat Electronic Warfare and Intelligence (CEWI)) (Corps). Washington, DC: Headquarters Department of the Army. Field Manual No. 34-20.

US Army (1984). Military Intelligence Battalion (Combat Electronic Warfare and Intelligence) (Aerial Exploitation) (Corps). Washington, D.C.: Headquarters Department of the Army, Field Manual No. 34-22.

VanCott, H.P. and Kinkade, R.G. (eds). (1972). Human Engineering Guide to Equipment Design, Revised Edition, U.S. Government Printing Office: Washington, D.C.

Vartabedian, A.G. (1972). Legibility of symbolism on CRT displays, Applied Ergonomics, 2, 130-132.

Vere S. (1983). Planning in Time: Windows and Durations for Activities and Goals. IEEE Transactions on Pattern Analysis and Machine Intelligence, 5(3), pp.246-266.

Warfield. R.W. and White, G.M. (1983). The new interface technology: An introduction to windows and mice, (Byte), 8(12) 218-230

Waterman, D. (1986). A Guide to Expert Systems Reading. MA Addison-Wesley.

Weizenbaum. J (1983) ELIZA–A computer program for the study of natural language communication between man and machine. Communications of the ACM, 26(1). 23-28

Wescourt .K. and Thorndyke, P. (1983) Alternative Knowledge Acquisition Interface Structures Defence Technical Information Center Technical Report. PPAFTR-1131-83-1. Defense Logistics Agency, Alexandria, VA:DTIC.

Winston. H. (1984). Artificial Intelligence. Reading, MA: Addison-Wesley.

Wixon, D., et al. (1983). Building a user-defined interface. Chi-83: Proceedings of the Conference on Human Factors in Computer Systems 24-27.

Woods, W. (1975) What's in a link. Foundations for semantic networks. Representation and Understanding: Studies in Cognitive Sciences, D. Bobrow, and A. Collins (Eds.). New York: Academic Press, pp. 35-82.

Yaghmai N., Maxin J., Expert Systems: A Tutorial, Journal of the American Society for Information Science, Vol. 35, No. 5, pp. 297-305.

Zachary. W. (1985). Beyond user-friendly: Building decision-aid interfaces for expert end users Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, 641-647.

Zachary, W., Glenn, F., Wherry, R. and Zaklad, A. (1981). Decision Situations, Decision Processes, and Decision Functions: Toward a Theory-Based Framework for Decision and Design. Paper presented at the First Conference on Human Factors in Computer Systems, National Bureau of Standards, Gaithersburg, MD.

Zadeh, Lofti A. (1984). Making Computers Think Like People. IEEE Spectrum, August, pp. 26-32.

Zadeh, Lofti A. (1985). Syllogistic Reasoning in Fuzzy Logic and its Application to Usuality and Reasoning with Dispositions. IEEE Transactions on Systems, Man, and Cybernetics. 15(6) November/December.

Zaklad. A., Zachary. W., Bulger. J., Glenn. F. and Hitchinbotham. J (1986). Decision Aids for the FAAD ABMOC. Technical Report 2018. Willow Grove. PA: Analytics.

APPENDIX C

EXTERNAL FILE INTERFACES - MMP1

THE FOLLOWING INFORMATION WAS PROVIDED BY MICRO ANALYSIS
AND DESIGN, INC. BOULDER, CO

APPENDIX C

External File Interfaces - MMP1

The following information was provided by Micro Analysis & Design, Inc., Boulder, CO

The data dictionaries specify the format of the data files which will be output by the SPREA. The information which is included in the specification consists of the following:

- the name of the file,
- a short textual description of the file contents, and
- the format of each record in the file.

NOTE: The first record of each file is an identification record. It contains information which will assist a programmer in retrieving the correct file for a given system description.

- a precise definition of each field of each record,
- the length of each field in the file, and
- an estimate of file length.

C-1 DATA DICTIONARY FORM

File Identification: System List

Description of Contents Lists the system description data for all systems for which there are Product 1 files.

Record	Field	Description	Length	Data Type
1		Identification record	80	Alphanum.
	1	Comment field	80	Alphanum.
2 - end		System description data	118	Alphanum.
	1	Mission area	50	Alphanum.
	2	System type	30	"
	3	System name	30	"
	4	Date last accessed	8	xx/xx/xx

Estimated Number of Records = 50?

Fixed or Variable Length File = Variable.

C-2 DATA DICTIONARY FORM

File Identification: System Missions

Description of Contents Lists the missions for a specific system which have Product 1 files.

Record	Field	Description	Length	Data Type
1		Identification record	118	Alphanum.
	1	Mission Area	50	Alphanum.
	2	System Type	30	"
	3	System Name	30	"
	4	Date Created	8	xx/xx/xx
2 - end		Mission Names	92	Alphanum.
	1	Mission number	12	"
	2	Mission name	80	"

Estimated Number of Records = 16.

Fixed or Variable Length File = Variable.

C-3 DATA DICTIONARY FORM

File Identification: Functions per Mission

Description of Contents: Lists the functions which are members of a specific mission.

Record	Field	Description	Length	Data Type
1		Identification record	42	Alphanum.
	1	System Name	30	Alphanum.
	2	Mission Number	12	"
2 - end		Function Names	93	Alphanum.
	1	Function number	12	"
	2	Function name	80	"
	3	Function type	1	"

Estimated Number of Records = 20.

Fixed or Variable Length File = Variable.

C-4 DATA DICTIONARY FORM

File Identification: Tasks per Function

Description of Contents Lists the tasks which are members of a specific function.

Record	Field	Description	Length	Data Type
1		Identification record	54	Alphanum.
	1	System Name	30	Alphanum.
	2	Mission Number	12	"
	3	Function Number	12	"
2 - end		Task Names	163	Alphanum.
	1	Task number	12	"
	2	Task name	80	"
	3	Decision type	1	"
	4	Following task no. 1	12	"
	5	Prob (task no. 1)	2	"
	6	Following task no. 2	12	"
	7	Prob (task no. 2)	2	"
	8	Following task no. 3	12	"
	9	Prob (task no. 3)	2	"
	10	Following task no. 4	12	"
	11	Prob (task no. 4)	2	"
	12	Following task no. 5	12	"
	13	Prob (task no. 5)	2	"

Estimated Number of Records = 8.

Fixed or Variable Length File = Variable.

C-5 DATA DICTIONARY ꜰOʀM

File Identification: Function Performance Criteria

Description of Contents Lists the performance time and accuracy criteria, as well as the accuracy standard, for each function in a given mission.

Record	Field	Description	Length	Data Type
1		Identification record	4	Alphanum.
	1	System Name	30	Alphanum.
	2	Mission Number	12	"
	3	Condition Set Number	12	"
2 - end		Function performance	266	Alphanum.
	1	Function number	12	"
	2	Time	xxxxx.xx	flt. pt. (min)
	3	Accuracy	xxx.xx	flt. pt (%)
	4	Accuracy standard	80	Alphanum.
	5	Comment	160	"

Estimated Number of Records = 8.

Fixed or Variable Length File = Variable.

C-6 DATA DICTIONARY FORM

File Identification: Function Accuracy Weighting

Description of Contents Lists the amount of weight assigned to each function accuracy in order to calculate mission accuracy (i.e., the probability of mission success).

Record	Field	Description	Length	Data Type
1		Identification record	54	Alphanum.
	1	System Name	30	Alphanum.
	2	Mission Number	12	Alphanum.
	3	Condition Set Number	12	"
2 - end		Task performance	296	Alphanum.
	1	Function number	12	"
	2	Accuracy Weight =	xxx.xx	flt. pt. (%)

Estimated Number of Records = 10.

Fixed or Variable Length File = Variable.

C-7 DATA DICTIONARY FORM

File Identification: Task Performance Criteria

Description of Contents Lists the performance time and accuracy criteria, as well as the accuracy standard, for each task in a given function.

Record	Field	Description	Length	Data Type
1		Identification record	66	Alphanum.
	1	System Name	30	Alphanum.
	2	Mission Number	12	Alphanum.
	3	Function Number	12	"
	4	Condition Set Number	12	"
2 - end		Task performance	296	Alphanum.
	1	Task number	12	"
	2	Time	xxxxx.xx	flt. pt. (min)
	3	Prob (level 0) Accuracy	xxx.xx	flt. pt (%)
	4	Prob (level 1) Accuracy	xxx.xx	flt. pt (%)
	5	Prob (level 2) Accuracy	xxx.xx	flt. pt (%)
	6	Accuracy std (level 0)	80	Alphanum.
	7	Accuracy std (level 1)	80	"
	8	Accuracy std (level 2)	80	"
	9	Prob (redo) (level 0)	xxx.xx	flt. pt (%)
	10	Prob (redo) (level 1)	xxx.xx	flt. pt (%)
	11	Prob (redo) (level 2)	xxx.xx	flt. t (%)

Estimated Number of Records = 10.

Fixed or Variable Length File = Variable.

C-8 DATA DICTIONARY FORM

File Identification: Corrective Maintenance Criteria

Description of Contents: Lists the maintenance ratios and MTTR by equipment by maintenance task.

Record	Field	Description	Length	Data Type
1		Identification record	66	Alphanum.
	1	System Name	30	"
	2	System Type	30	"
2 - end		MRs and MTTRs	266	Alphanum
	1	Equipment Type	20	"
	2	Overall MR	xxx.xx	flt. pt.
	3	Overall MTTR	xxxxx.xx	"
	4	Inspection ORG MR	xxx.xx	"
	5	Inspection ORG MTTR	xxxxx.xx	"
	6	Inspection DS MR	xxx.xx	"
	7	Inspection DS MTTR	xxxxx.xx	"
	8	Inspection GS MR	xxx.xx	"
	9	Inspection GS MTTR	xxxxx.xx	"
	10	Repair ORG MR	xxx.xx	"
	11	Repair ORG MTTR	xxxxx.xx	"
	12	Repair DS MR	xxx.xx	"
	13	Repair DS MTTR	xxxxx.xx	"
	14	Repair GS MR	xxx.xx	"
	15	Repair GS MTTR	xxxxx.xx	"
	16	Replace ORG MR	xxx.xx	"
	17	Replace ORG MTTR	xxxxx.xx	"
	18	Replace DS MR	xxx.xx	"
	19	Replace DS MTTR	xxxxx.xx	"

Cont.

Record	Field	Description	Length	Data Type
	20	Replace GS MR	xxx.xx	flt. pt.
	21	Replace GS MTTR	xxxxx.xx	"
	22	Test ORG MR	xxx.xx	"
	23	Test ORG MTTR	xxxxx.xx	"
	24	Test DS MR	xxx.xx	"
	25	Test DS MTTR	xxxxx.xx	"
	26	Test GS MR	xxx.xx	"
	27	Test GS MTTR	xxxxx.xx	"
	28	Troubleshoot ORG MR	xxx.xx	"
	29	Troubleshoot ORG MTTR	xxxxx.xx	"
	30	Troubleshoot DS MR	xxx.xx	"
	31	Troubleshoot DS MTTR	xxxxx.xx	"
	32	Troubleshoot GS MR	xxx.xx	"
	33	Troubleshoot GS MTTR	xxxxx.xx	"
	34	Troubleshoot Accuracy	xxx.xx	"

Estimated Number of Records = 13.

Fixed or Variable Length File = Variable.

C-9 DATA DICTIONARY FORM

File Identification: System RAM Criteria

Description of Contents: Lists the reliability, availability, and maintainability criteria for the system.

Record	Field	Description	Length	Data Type
1		Identification record	66	Alphanum.
	1	System Name	30	"
	2	System Type	30	"
	3	Scenario Comment	80	"
2		Availability	6	Alphanum.
	1	Operational Availabilty	xxx.xx	(%)
3		Maintainability	14	
	1	Maintenance Ratio	xxx.xx	hr/op hr
	2	MTTR	xxxxx.xx	hours
4		Reliability		
	1	Mobility	xxxxx.xx	
	2	Measure	5	miles/km/flthr
	3	Usage (Daily)	xxxxx	
5		Reliability		
	1	Armaments	xxxxx.xx	
	2	Measure	5	rnds
	3	Usage (Daily)	xxxxx	
6		Reliability		
	1	Communication	xxxxx.xx	
	2	Measure	5	hours/mins
	3	Usage (Daily)	xxxxx	

Estimated Number of Records = 6.

Fixed or Variable Length File = Fixed.

C-10 DATA DICTIONARY FORM

File Identification: Condition Set

Description of Contents: Lists the Condition Settings for a specific condition set number.

Record	Field	Description	Length	Data Type
1		Identification record	92	Alphanum.
	1	Condition set name	80	Alphanum.
	2	Condition set number	12	"
2	1	Environmental (Basic)	120	Alphanum.
3	1	Terrain (Basic)	20	Alphanum.
4	1	Target/Threat (Basic)	120	Alphanum.
5	1	Friendly Force (Basic)	120	Alphanum.
6	1	Environmental (Add'l)	120	Alphanum.
7	1	Terrain (Add'l)	120	Alphanum.
8	1	Target/Threat (Add'l)	120	Alphanum
9	1	Friendly Force (Add'l)	20	Alphanum.

Estimated Number of Records = 9.

Fixed or Variable Length File = Variable.

APPENDIX D

EXTERNAL FILE INTERFACES - MMP5S

THE FOLLOWING INFORMATION WAS PROVIDED BY SAIC

APPENDIX D

External File Interfaces - MMP5-S

The following information was provided by SAIC

The data dictionaries specify the format of the data files which will be output by the MMP5-S. The information which is included in the specification consists of the following:

- the name of the file,
- a short textual description of the file contents, and
- the format of each record in the file.

NOTE: The first record of each file is an identification record. It contains information which will assist a programmer in retrieving the correct file for a given system description.

- a precise definition of each field of each record,
- the length of each field in the file, and
- an estimate of file length.

RECORD/Field Name	Type/Prec.	Range	U/M	Comments
SYSTEM_TYPE				
• System_type_id	I/2	1-21	n/a	
System_type_nm	C/30	n/a	n/a	e.g., attack helicopter "._nm" corresponds to "name"
SYSTEM_TYPE-FUNCTION_TASK				
• System_type_id	I/2	1-21	n/a	
• Func_task_id	I/4		n/a	
FUNCTION_TASK				
• Func_task_id	I/4		n/a	Functions and tasks share the same id attribute
Func_task_nm	C/30	n/a	n/a	e.g., operational, plan and prepare for mission, plan flight
PARENT_FUNCTION-TASK_SEQUENCE				
• Func_task_id	I/4		n/a	
• Parent_func_id	I/2	n/a	n/a	Notion of hierarchies of funcs.
Task_sequence	I/2		n/a	Only for lowest level funcs., and tasks
SYSTEM_TYPE-EQUIPMENT				
• System_type_id	I/2	1-21	n/a	
• Equipment_id	I/3		n/a	Generic; requires qualification w/System type
EQUIPMENT				
• Equipment_id	I/3		n/a	
Equipment_nm	C/30	n/a	n/a	e.g., Power plant, engine system
Parent_equip_id	I/3		n/a	Notion of hierarchies of equip.
CONDITION				
• Cond_nm	C/30	n/a	n/a	e.g., Precipitation
CONDITION_VALUE				
• Cond_value_nm	C/30	n/a	n/a	e.g., Rain
Cond_nm	C/30		n/a	e.g., Precipitation

RECORD/Field Name	Type/Prec.	Range	U/M	Comments
CONDITION_VALUE				
• Cond_value_nm	C/30	n/a	n/a	e.g., Rain
Cond_nm	C/30		n/a	e.g., Precipitation
PERFORMANCE_OBJECTIVE				
• Perf_obj_id	I/4		n/a	
Perf_obj_nm	C/30	n/a	n/a	e.g., Plan and prepare for mission
PERFORMANCE OBJECTIVE-CONDITION_UTILIZED				
• Perf_obj_id	I/4		n/a	
• Cond_value_nm	C/30	n/a	n/a	e.g., Rain
Perf_crit_type	I/1	1-3	n/a	corresponds to Category 1, 2, and 3 type tasks

Perf_crit_value	I/6			for type 1: secs
				for type 2: times/sec
				for type 3: secs

CONDITION UTILIZED				
• Cond_value_nm	C/30	n/a	n/a	e.g., Rain
SYSTEM_COMPONENT				
• System_comp_id	I/3	n/a	n/a	"instance" of system type/equip
System_type_id	I/2	1-21	n/a	
Equipment_id	I/3	n/a	n/a	
SYSTEM_COMPONENT-TASK_UTILIZED				
• System_comp_id	I/3		n/a	"instance" of system type/equip
• Task_id	I/4		n/a	
Cluster_id	I/3		n/a	Derived by Cluster Tasks process
From_node				Defines Precedence Network link
F_System_comp_id	I/3		n/a	
F_task_id	I/4		n/a	
To_node				Defines Precedence Network link
T_System_comp_id	I/3		n/a	
T_task_id	I/4		n/a	
Job_id	I/3		n/a	Derived by Derive Unique Jobs process

RECORD/Field Name	Type/Prec.	Range	U/M	Comments
TASKS-UTILIZED				
• Task_id	I/4		n/a	
Task_nm	C/30	n/a	n/a	Not necessarily from K/C Tax
PREDECESSOR_TASK				
• Task_id	I/4		n/a	
Predcssor_task_id	I/4		n/a	
SUCCESSOR_TASKS				
• Task_id	I/4		n/a	
• Successor_task_id	I/4		n/a	
PERFORMANCE_OBJECTIVE-CONDITION/SYSTEM_COMPONENT_TASK_MATRIX				
• System_comp_id	I/3		n/a	
• Task_id	I/4		n/a	
• Perf_obj_id	I/4		n/a	
• Cond_value_nm	C/30	n/a	n/a	
Binary_value	I/1	0,1	n/a	Element of matrix
TASK_TIME				
• System_comp_id	I/2	1-21	n/a	
• Equipment_id	I/3		n/a	
• Task_id	I/4		n/a	
• Cond_value_nm	C/30	n/a	n/a	
Task_time	I/6	n/a	secs	
CLUSTER				
• Cluster_id	I/3		n/a	
Parent_cluster_id	I/3		n/a	Notion of hierarchy of clusters
Category	I/1	1-3	n/a	Derived from Per_crit_type of PERFORMANCE_OBJECTIVE-CONDITION_UTILIZED
JOB				
• Job_id	I/3		n/a	

APPENDIX E

REVIEW AND IMPLICATIONS OF TIME AND MOTION

Systems for the Development of SEPTA

Robert J. Wherry, Jr., Ph.D.

E-1. INTRODUCTION

The Human Operator Simulator (HOS) represents a unique approach for predicting the performance of humans in complex human-machine systems. This simulation approach has been successfully developed and tested over a period of twenty years. However, the prior version, HOS III, was programmed to simulate stationary, seated, operators whose tasks involved the reading of displays and the manipulation of control devices. It is necessary for the revision (HOS IV) to be able to simulate persons whose tasks require them to move about their workstations and to pick up and move objects from one place to another. This additional capability would represent a considerable advance in simulation technology, and would enable the new programs to be applicable for analyzing the performance of maintenance and handling personnel as well as operators who must occasionally move about a workstation to handle various types of objects.

E-1.1 Objectives

A major purpose of this document was to review various ways in which human acts related to locomotion and object handling have been described by "Time and Motion" systems.

A related purpose of this document was to provide background for the development of HOS IV micromodels based on the review of the T&M systems.

E-1.2 Review of Time and Motion Systems

Because industrial applications of Time and Motion (T&M) systems have long been concerned with workers in plants who also must sometimes move from place to place and must also often manipulate objects, an in-depth review of the T&M systems appeared warranted. The T&M review presented in this document provides a brief history of T&M systems as well as a review of what are considered by T&M systems as the basic "elemental motions" required of workers.

E-1.3 HOS IV Design Guidance

Addition of locomotion and object handling capabilities for HOS IV will necessitate development of:

- a basic vocabulary, task-describing language for movement,
- the structure of a general-purpose human model that includes the micromodels of HOS that decide what actions the simulated operator will attempt, and
- the general-purpose "strategy" procedures that simulate an intelligent, adaptive human.

While these considerations can be individually discussed, they are, in fact, highly interrelated since the words used to describe tasks must be able to be "parsed" and "understood" by the modeled human. Further, the goals of task statements must ultimately be translated into simulated actions in the context of dynamically changing situations.

E-1.3.1 New Language Requirements

The T&M review is particularly useful in beginning to understand the "language" requirements for the advanced advanced system. The language permits the HOS user to describe what is expected of the human to be simulated by describing the tasks to be done in English-like statements. Because earlier HOS versions did not consider operator locomotion or object manipulation tasks, words to express these types of actions are not present in the current interpreter. The vocabulary of the models must be augmented with new types of human action words to permit appropriate and unambiguous task descriptions.

E-1.3.2 The Structure of the HOS IV Human Model

To a large extent, a major purpose of these efforts was to understand the needed framework for the HOS IV Human Model. A human model not only contains micromodels for the "elemental" actions that can be carried out by the simulated human, but also many procedural strategies for deciding how to go about prosecuting various assigned tasks. The reader of this document will observe that the concept of satisfying goals is central to the HOS approach and, therefore, to the framework for the human model. Statements written in the simulation language must be able to be "understood" by the human model as goals to be accomplished. This means that a portion of the human model is a "statement handler" that can decide what resident procedures and/or what modeled microactions must be invoked to satisfy the desired task goals intelligently. The reviews and discussions of various types of needed acts is, thus, preliminary to deciding upon the necessary and desired human model framework for the advanced HOS.

E-1.3.3 Procedures for HOS IV

As mentioned above, the human model must contain strategies, rules, and procedures that permit the simulated operator to prosecute and perform assigned tasks. While some strategies, rules, and procedures are specific to a given job, many others are sufficiently general that they should be written, tested, and furnished as part of the human model. An example of a highly general procedure or strategy would be one needed to decide what task to work on in situations in which the simulated human has multiple ongoing task responsibilities. Another general strategy procedure would allow one to decide how to go about getting particular information that is needed by a task statement (i.e., deciding if the simulated human will attempt to recall the information or to find it in the external world). While previous versions of HOS already contain several such resident procedures, the addition of locomotion and object manipulation tasks introduces new requirements for new general procedures.

E-1.4 Organization of This Report

The historical backgrounds of both T&M systems and HOS are discussed in Section E–2. The review of T&M elemental motions is presented in Section E–3.

Detailed discussions are presented for whole-body movements in Section E–4, object handling in Section E–5, mental calculations and computations in Section E–6, and visual/perceptual actions in Section E–7.

Conclusions about the feasibility of including the capability for locomotion and object handling in HOS IV are discussed in Section E–8 along with recommendations for those developments.

E-2. EARLY DEVELOPMENT OF TIME AND MOTION (T&M) ANALYSIS

By the start of the twentieth century scientific enquiry into the times required to perform various acts already had a long history. Reaction time (RT) studies had been conducted since the early 1860s by Hirsh for the eye, ear, and touch. Donders in 1868 had attempted to measure the times to do various mental processes. Wundt opened his laboratory in Leipsig in 1879 and he and his students continued RT studies. In the field of vision, Javal had noted in 1878 that reading was composed of a series of discrete fixations rather than a continuous sweep. Dodge had invented a photographic method for recording and measuring eye movements and in 1901 he and Cline had reported various eye movement times.

In applied settings, "time studies" (i.e., finding the average time to make a certain object) can be traced back to M. Perronet in France in 1760 and later to Charles Babbage in England in 1830. In the United States, Frederic Taylor is credited with conducting the first "time studies" in 1881 at a steel company in Philadelphia. Application of scientific T&M studies to manual operations in industrial settings is generally acknowledged to have started with the work of Frank and Lillian Gilbreth in 1911 and Fredrick W. Taylor in 1912. What was distinctly different about T&M studies was the ultimate recognition that the time needed to perform a particular factory job was dependent on the motions required by the job.

In 1885, at the age of seventeen, Frank Gilbreth got a job as a brick layer. He observed that each brick layer had his own "style" of laying bricks, and that many of those styles were terribly inefficient. He later demonstrated to the foreman that great savings in time could be realized by "standardizing" the manner of arranging the materials, applying the mortar, and laying the bricks. He urged "training programs" for novice brick layers so that they could learn the "best way" to lay bricks. The experience of discovering the most "economical" motions for a given job by analyzing what had to be done was highly rewarding and dominated the rest of his professional career. Gilbreth's original definition of "motion study" was "dividing work into the most fundamental elements possible; studying these elements separately and in relation to one another; and from these studied elements, when timed, building methods of least waste" (Gilbreth, F. & Gilbreth, L., 1917). The Gilbreths developed seventeen shorthand symbols called "therbligs" ("Gilbreth" backwards) to record operator sequences of using "fundamental motion elements." They quickly saw the utility of using motion pictures to record and analyze motions on a frame-by-frame basis. This technique was presented at an annual meeting of the American Society of Mechanical

Engineers in 1912, and they became, at least for industrial work settings, the pioneers of "micromotion" analysis.

Taylor's contributions to the applied study of human work in industry were also immensely significant. In his role as production supervisor in an industrial setting he also observed that much time and effort was wasted by the idiosyncratic ways that workers did their jobs and the almost total lack of specific goals. He concluded that, "The greatest production results when each worker is given a definite task to be performed in a definite time and in a definite manner." With this single sentence, Taylor anticipated the need for job descriptions (i.e., "a definite task") as well as performance criteria specifications (i.e., the specific actions by which tasks should be accomplished and the time allocated to perform those tasks). Another of Taylor's major contributions were "wage incentives" for workers who exceeded the "standard times" established for a given task. Taylor's method of determining standard times for various tasks consisted, for the most part, of using a stop-watch to measure experienced workers performing those tasks. Taylor also studied "work/rest" cycles to overcome fatigue built up by various types of physical work.

Most of the earliest T&M studies were accomplished in the context of factory work with emphasis being given to developing more efficient (economical) work procedures. The concept of developing tables of predetermined times for various actions and motions was first initiated by A. B. Segur around 1919. Segur had been particularly struck by his observation that experienced workers were both faster and less variable than inexperienced ones. Through extrapolation of those observations, he formulated what came to be known as "Segur's Law." It stated, "Within practical limits the times required of all expert workers to perform true fundamental motions are constant." Today, his assertion is recognized as being unwarranted; ample evidence exists for stable individual differences even after extensive practice. However, by assuming it to be true, predetermined times for all basic motions appeared feasible. Segur's Motion-Time Analysis (MTA) system was thus the first of the T&M systems containing tables of predetermined times for various motions. By 1925, Segur's MTA system was fairly well developed.

Another group of industrial engineers, headed by J. H.Quick, independently developed the "Work Factor Time Standards" (WFTS) system in the middle 1930s. Their system attempted to account for the times of basic motions by a relatively few "workfactors." Those considered as major determiners of time required for various kinds of motions were:

- the particular body member (e.g., finger-hand, arm, forearm swivel, trunk, leg, and foot) involved in the basic motion,
- the distance moved by the body member,

- the weight (or resistance) of the object being manipulated, and
- the extent of manual (i.e., muscular, visual, and mental) control needed.

The first three work factors could be determined (i.e., measured) in an objective fåshion. However, the fourth work factor required estimating the extent of "manual control" for each motion:

- if the motion included a definite stop,
- if the motion had to be aimed,
- if the motion included a change in direction, and
- if caution must be exercised during the motion.

The final factor, thus, introduced subjective judgement by the user of the WFTS system. Further, the extent of manual control needed was to be judged as being "low", "medium", or "high", rather than on a continuum.

It is readily accepted that the amount of muscular, visual, and mental control needed for any task will determine, to a large extent, the difficulty of that task, and, consequently, how long it takes. It should also be recognized that determining the extent of this "work factor" for a given motion is not dissimilar to determining what today might be referred to as estimating the extent of "motor", "perceptual", or "cognitive" workload for a particular task. Thus, while the term "workload" is a relatively new concept in human factors, the need to consider its importance can also be traced back to early T&M systems.

In the early developmental years for T&M systems, bitter disagreements arose over whether one should be primarily concerned with "time studies" or "motion studies" as well as where and how they should be conducted. Some investigators were only interested in "practical" time studies in "real-world" applied settings; others favored carefully controlled, scientific motion studies conducted in a laboratory. Those familiar with human factors research can readily appreciate these types of disputes.

Over the years, the earliest T&M systems were modified and improved. Newer ones were also developed, but most represented only minor adaptations of the original concepts. In 1948, the Methods-Time Measurement (MTM) system was developed by H. Maynard, G. Stegemarten, and J. Schwab. While it, too, has undergone modifications, it remains the most widely used of the "predetermined" time and motion systems. In addition to objectively measurable parameters such as distance and object weight, it also requires usage of "leveling" procedures that utilize subjective judgements of the "skill" and "effort" required for given motions.

The rationale (and validity) of T&M systems can be seen to be based on the following assumptions:

- manual operations, even complex ones, during an operator's work are composed of "basic motions",
- complex work action sequences can be analyzed into their "basic motions",
- time to complete a "basic motion" is determined by the nature of the motion and the conditions under which the work is performed,
- the "standard" times to complete various basic motions for various conditions can be established, and
- the total time to complete various work action sequences will be the sum of the times for the basic motions.

A major similarity of almost all T&M systems deals with how predetermined times are presented. Each system presents its version of the "elemental" motions. These, in turn, are usually divided into subcategories that represent different levels of difficulty that are brought about by a few major attributes of those motions (e.g., the weight of an object, the distance it must be moved, the care that must be exercised, etc.). The various attributes are typically divided into three or four "discrete" levels (rather than a "continuum"). Because of this, it was possible for the T&M systems to assign specific "predetermined" times to each possible subcategory of the elemental motions. Further, because there are relatively few attributes and subcategories, the predetermined times could be presented in tabular form in fairly simple tables. These tables could then be used by T&M practitioners to "look up" appropriate times once the "motions" for a job had been described. Summing the "elemental motion" times would then yield the overall time required or allocated to perform the job.

The T&M approach of a data base for storing standard times to perform certain types of actions thus also anticipated various human factors data bases such as Swain and his colleague's AIR Data Store.

Among the applicable goals that can be satisfied, according to the MTM authors, by appropriate use of time and motion systems are:

- developing effective methods prior to production
- improving existing methods
- establishing time standards

- developing time formulae or standard data
- estimating production costs
- guiding product design
- developing effective tools
- selecting effective equipment
- training employees to be highly methods conscious
- setting wage rates
- settling grievances, and
- accomplishing other needed research.

These goals, while certainly aimed at providing a "fair and equitable" means for compensating the labor force, were also obviously aimed at satisfying management goals of "reducing costs of production and operations." Many of them can also be seen as being identical to human factors goals of improving human performance through job design, training programs, workplace layout, selection of devices with which the human element must interface, and so forth. Thus, it is fair to say that there should be a great deal of overlap between T&M systems and many human factors techniques.

E-3. REVIEW OF BASIC ELEMENTAL MOTIONS

A major objective of this report was to identify the "elemental" actions required for describing human locomotion and object manipulation. Because a large portion of the activity of factory workers in the 1930 to 1960 period consisted of actions requiring locomotion and object manipulation, it seemed appropriate to review several of the time and motion systems since they claimed their results to be the outcome of scientific studies of such actions.

Four different time and motion systems were reviewed. The titles, acronyms and references for the systems reviewed are shown below:

- Methods-Time Measurement (MTM) (Maynard, H. et al., 1948) (Karger, D. and Bayha, F., 1957; 1966) (Antis, W. et al., 1963; 1968)
- Motion Economy & Work Measurement (MEWM) (Morrow, R., 1946; 1957)
- Master Standard Data (MSD) (Crossan, R. and Nance, H., 1962)
- Work Factor Time Standards (WFTS) (Quick, J. et al., 1962).

While these systems categorize their "basic actions" in a variety of different ways, comparison of those basic actions led to a new taxonomy that could cover all four systems. The taxonomy of basic motions/actions include five major categories and several subcategories as follows:

- whole body actions
  - relocation and turning
  - stance-changing (w/o major relocation)
- lower-limb actions
- upper-limb actions
  - object-obtaining/holding/releasing actions
  - object-positioning/aligning/orienting actions
  - miscellaneous object-manipulating macro-actions
  - multi-object actions
- mental "decision/computation" actions
- visual/perceptual actions
  - simple eye-movements
  - miscellaneous visual decision-making tasks.

The basic "elemental" actions/motions and their applicable definitions, as found in the four time and motion systems reviewed, are described in the sections that follow.

E-3.1 WHOLE BODY RELOCATION AND TURNING ACTIONS:

WALK = "a forward or backward movement of the body performed by alternate steps" (MTM)

WALK = "relocation of more than 30 inches" (WFTS)

PLAIN OLD-FASHIONED WALKING = "gets us from one place to another" (MSD)

TURN BODY = "rotational movement of body performed by one or two steps" (MTM)

BODY TURN = "trunk is rotated by movement of one or two feet" (WFTS)

IN-PLACE WALKING = "turning" (MSD)

SIDE STEP = "a lateral movement of body, without rotation, performed by one or two steps" (MTM)

MINOR RELOCATION = "30 inches or less" (WFTS)

UNCONSCIOUS WALKING = "moves us about the place at which we already are; one or both feet are used to move forward, backward, sideward, or around the axis of the body" (MSD)

CLIMB & DESCEND STAIRS (WFTS)

E-3.2 STANCE-CHANGING ACTIONS (WITHOUT MAJOR RELOCATION):

BODY TWIST = "trunk twists while feet are fixed" (WFTS)

BEND = "motion of lowering the body in a forward arc from standing position so that hands can reach to or below the level of the knees" (MTM)

BEND = "hand is taken to position about half-way between knee and floor" (MSD)

STOOP = "motion of lowering the body in a forward arc from a standing position so that hands can reach the floor" (MTM)

STOOP = "like BEND, but knees are bent and hands are lowered an additional 5 or 6 inches" (MSD)

KNEEL = "motion of lowering body from erect standing position by shifting one foot forward or backward and lowering one knee to floor (other knee may also be placed on floor adjacent to first knee)" (MTM)

KNEEL = "if operator wants to stay in a floor level position for some time, knee is rested on floor so that operator will not become fatigued" (MSD)

SIT = "motion of lowering body from erect standing position directly in front of seat and transferring weight of body to seat" (MTM)

STAND = "motion of transferring weight from seat and raising body to erect standing position directly in front of seat" (MTM)

E-3.3 LOWER-LIMB ACTIONS (WITHOUT BODY RELOCATION):

LEG MOTIONS = "movement of the leg in any direction with the knee or hip as a pivot, where the predominant purpose is to move the feet rather than the body" (MTM)

FOOT MOTIONS = "movement of ball of foot up or down with the heel or instep as the fulcrum" (MTM)

E-3.4 UPPER-LIMB OBJECT-OBTAINING/HOLDING/RELEASING ACTIONS:

REACH = "basic hand or finger motion employed when predominant purpose is to move hand or fingers to a destination" (MTM)

TRANSPORT REACH = "primary purpose of body member motions is to relocate body member (to objects)" (WFTS)

TRANSPORT REACH = "moving hand with nothing in it" (MEWM)

GRASP = "basic finger or hand element employed when to secure control of an object" (MTM)

GRASP = "act of obtaining control of one or more objects" (WFTS)

GRASP = "gaining control of an object (begins when hand/finger contacts object)" (MEWM)

OBTAIN = "the basic element by which one gains control of one or more objects so that the next basic element can be performed" (MSD)

HOLD = "time during which a body member functions as a vise" (WFTS)

HOLD = "retaining an object in a fixed position without moving it" (MEWM)

MOVE = "basic hand or finger motion employed when predominant purpose is to *transport an object to a destination*" (MTM)

TRANSPORT MOVE = "primary purpose of body member motions is to relocate object" (WFTS)

TRANSPORT LOADED = "a part is carried, slid, dragged, or pushed by the hand" (MEWM)

PLACE = "basic element used to move something to an immediate destination" (MSD)

RELEASE = "basic finger or hand motion employed to relinquish control of an object" (MTM)

RELEASE = "act of body member separating itself from an object" (WFTS)

RELEASE LOAD = "palm turned down, opening of the hand with an object in it" (MEWM)

E-3.5 UPPER-LIMB OBJECT-POSITIONING/ALIGNING/ORIENTING:

POSITION = "basic finger or hand element employed to align, orient, and engage one object to another to obtain a specific relationship" (MTM)

PRE-POSITION = "act of turning and orienting an object to a correct position for a subsequent element of work" (WFTS)

PRE-POSITION = "any movement to place an object so that it will be handiest for use the next time it is needed" (MEWM)

FINGER SHIFT = "a series of minor movements of the fingers performed for the purpose of pre-positioning an object prior to performing the basic element place" (MSD)

ALIGN = "consists of all basic motions required to make coincident the insertion axes of the engaging and engaged parts during a position" (MTM)

ORIENT = "all basic motions required to geometrically match the cross section of engaging part and that of engaged part about axis of insertion" (MTM)

POSITION = "to line up one object with another" (MEWM)

ROTATE BY GRASP = "turning object by grasping object with fingers and rotating fingers and/or wrist" (MSD)

TURN = "basic motion employed to rotate the hand about the long axis of the forearm" (MTM)

CRANK = "motion of the fingers, hand, wrist, and forearm in a circular path with the forearm pivoting at the elbow" (MTM)

ROTATE BY CRANK = "turning object by using crank" (MSD)

APPLY PRESSURE = "application of a muscular force to overcome object resistance, accompanied by little or no motion" (MTM)

APPLY PRESSURE = "application of pressure in which no motion occurs but during which the muscles tense or flex" (WFTS)

EXERT FORCE = "an instantaneous application of more than normal force applied for the purpose of overcoming initial resistance or forcing an object against another" (MSD)

E-3.6 MISCELLANEOUS UPPER-LIMB OBJECT-MANIPULATION "MACRO" ACTIONS:

PICK-UP = "combination of REACH, GRASP, and MOVE" (WFTS)

PLACE-ASIDE = "combination of MOVE, RELEASE, and MOVE" (WFTS)

E-3.7 UPPER-LIMB "MULTI-OBJECT"-RELATED ACTIONS:

ENGAGE = "insertion of engaging part into engaged part following completion of ALIGN and ORIENT" (MTM)

ASSEMBLE = "act of joining two or more objects together" (WFTS)

USE = "act of working with a tool or other device, or of using one or more body parts as a tool" (WFTS)

ASSEMBLE = "putting parts together" (MEWM)

USE = "placing a positioned object in or on another object with which it forms an integral part or in manipulating a tool or device in a manner for which it was intended" (MEWM)

USE = "movements of the fingers, hand, and/or arm where motions are not limited to any direction or any place" (MSD)

DISENGAGE = "basic hand or finger element employed to separate one object from another where there is a sudden ending of resistance" (MTM)

DISASSEMBLE = "act of separating objects that have been previously joined together" (WFTS)

DISASSEMBLE = "taking a part away from another part" (MEWM)

BALANCING DELAY = "short periods of time during which one hand (or operator) waits for other to complete its motions" (WFTS)

DELAY = "occurs to an object when conditions, except those which intentionally change the physical or chemical characteristics of the object, do not permit or require immediate performance of next planned action" (MEWM)

STORAGE = "occurs when object is kept and protected against unauthorized removal" (MEWM)

E-3.8 MENTAL "DECISION/COMPUTATION" ACTIONS:

PLAN = "deciding how to proceed with the work" (MEWM)

SELECT = "the choice of one object from others" (MEWM)

COMPUTE = "add, subtract, multiply, and divide" (WFTS)

E-3.9 VISUAL/PERCEPTUAL ACTIONS:

EYE TRAVEL = "basic eye motion employed to shift the axis of vision from one location to another" (MTM)

EYE SHIFT = "time required to shift line of sight from point to point and fixate the eyes for clear vision at the end of the shift" (WFTS)

EYE FOCUS = "basic visual and mental element of looking at an object long enough to determine readily distinguishable characteristics" (MTM)

EYE FOCUS = "time required for eye muscles to accommodate (alter lens shape, adjust pupils, fixate eyes) in order to bring into sharp focus an object which can be clearly seen in perfect detail" (WFTS)

SEARCH = "occurs when eye looks for one object among several, or when the hand moves around to locate an object" (MEWM)

FIND = "follows a SEARCH" (MEWM)

INSPECT = "a visual process for determining the presence, absence, quality, quantity, or identity of any one or group of characteristics" (WFTS)

INSPECTION = "occurs when an object is examined for identification or is verified for quality or quantity in any of its characteristics" (MEWM)

INSPECT = "consists of a series of visual pauses (where eye registers impression) and visual travel (where eye sweeps over material without regarding detail)" (MSD)

READING = "occurs as a series of eye travels and eye focuses" (MTM)

ACTION READ = "identification of letter symbols, words, numbers, and instructions as required for recognition to permit appropriate reaction" (WFTS)

CONCEPT READ = "read operations involving relatively long passages and requiring the operator to understand and remember the concepts and principles stated in the material, but not necessarily to recall details" (WFTS)

VISUAL CHECKING = "commonly associated with measurements (i.e., checking dimensions with ruler)" (MSD)

VISUAL COMPARISON = "involves comparing numbers or symbols (series of visual pauses)" (MSD)

REACT = "operator's act of receiving external messages or signals (stimuli) and preparing to perform the appropriate subsequent physical or mental acts" (WFTS)

E-4. "GOAL-SATISFYING" APPROACH FOR WHOLE-BODY MOVEMENTS

While there were obvious similarities among the four reviewed T&M systems, Section E-3 pointed out disagreement over what was included as the set of basic actions or elemental motions. It was also obvious that all of the systems recognized the necessity for including some perceptual and cognitive actions as well as motor actions (i.e., motions) in order to account for time required to perform some task. Thus, the time and motion systems were, in reality, time and action systems that have emphasized the importance of motor actions. This was understandable considering that developers of the T&M systems were interested in accounting for the time for factory employees to perform their jobs. Factory work, especially in the early part of this century, was dominated by physical rather than mental activities. Even more disturbing was that, even when one only considered the motor actions of the T&M systems, they did not appear to be elemental ones. Instead, an odd mixture of relatively simple (perhaps, elemental) and complex actions were found. Some motions involved only a single limb and could be performed in a brief period of time; others involved many parts of the anatomy, required repetitive motions, and could require an extended period of time to complete.

The definitions for the elemental motions/actions appeared, at times, ambiguous. Some definitions were couched in terms of the anatomy involved while others stressed the purpose of the action. Finally, it was not readily apparent why many additional or alternative actions had not been included.

The following sections discuss an alternative means of describing tasks and jobs and the activities required for various jobs. This approach, which makes use of "resident general procedures," can be referred to as a goal-satisfying approach to simulating human performance. As will be seen, the resident procedures make extensive use of pre-learned strategies, algorithms, and rule-based decisions. This approach has been successfully used in previous versions of the Human Operator Simulator (HOS III). However, HOS III did not permit locomotion of the operator. Because it is desired to include locomotion in HOS IV, the elemental actions for whole-body movement are of particular interest and are discussed in this section.

E-4.1 TRAVEL TO Specified-place

It is difficult to defend walking as an elemental motion since it obviously is a fairly complex series of actions. Normal walking involves continuous and repeated executions of coordinated, forward-moving, alternating steps by the left and right lower-limb parts of the body. Further, upper-limb motions as well as those of the trunk, torso, neck, and head may also be present during walking to maintain whole-body balance. It is, perhaps, preferable to consider walking not as an elemental motion, but as a goal-satisfying activity. Walking can be thought of as a subroutine or procedure that may be invoked whenever the human needs to move his body to a new place that is greater than a certain distance away from his current location Thus, walking is a means to an end. It is simply one method by which a person moves their whole-body to a new location.

E-4.1.1 Alternative Methods of Travel

There are, however, several alternative methods (e.g., running, jumping, leaping, hopping, skipping, or even crawling) that were not listed in any of the T&M systems. These motions could also be used to accomplish the same objective for which walking was included. Of course, it is possible that alternative travel actions were, at one time, considered by the T&M developers, but were excluded as being undesirable actions to impose on workers. Nevertheless, even though walking may be the usual preferred method for travelling to a new place, at least some of the alternative methods might be usefully invoked under some conditions to accomplish the same goal for which walkinq would be done. Some emergency conditions, for instance, may dictate (or at least encourage) that travel be accomplished as rapidly as possible.

E-4.1.2 Factors Affecting the Choice of Travel Methods

The choice between walking or running (or other alternative travel methods) to a new place, as well as how fast those activities will be carried out, are undoubtedly determined by a number of different factors. Some of the factors would include: the perceived time available for accomplishing the relocation, the distance to be travelled, the particular load (if any) that is being carried by the person, the presence of obstacles along the path to be travelled, and the conditions of the surface over which the person will be travelling. It should be noted that many of these factors are similar to those mentioned earlier as work factors. Thus, an individual can decide which alternative to select on the basis of rules.

E-4.1.3 Vertical Travel Methods

Climbing and descending stairs or ladders are also travel activities that are not dissimilar to walking. These activities are invoked when the path being travelled includes stairs, ramps, ladders, etc. Climbing or descending a steep ladder (or stairway) may require heavier involvement of the upper-limbs. Again it is possible that the person who is in a hurry will choose to run (rather than walk) up stairs that are not overly steep. Thus, the angle of incline of the staircase, ramp, or ladder, the size and spacing of individual stairs or rungs, and the availability of handrails, in addition to the previously mentioned traveling factors, will determine the particular strategy chosen by the person during periods of climbing and descending.

E-4.1.4 Non-forward Horizontal Travel Methods

The decision to travel sideways (i.e., MTM's side step) rather than to travel forward is a useful distinction in that the lower-limb motions involved are somewhat different than those in normal walking. MSD's unconscious walking also covered small leg and foot movements in any direction (forward, sideways, backwards, etc.) as did WFTS's minor relocation. However, walking backwards for distances greater than 30 inches does not appear to be covered by any of the T&M systems. Some workplace layouts and some conditions may make that type of movement necessary (e.g., where a heavy load must be pulled or dragged through a narrow passage, or after work has to be accomplished at the end of a narrow cul de sac).

E-4.1.5 Horizontal Reorientation during Travel

Orienting the body to face in a new X-Y direction (e.g., Body turn, Turn body, and In-place walking) may be required both while not engaged in travel and when moving to a new place. Reorienting the body while travelling is necessary to negotiate a path that has one or more turns in it, to avoid obstacles in a path, and to be able to end the travelling in a favorable position for carrying out further actions for which the travel is being accomplished. This emphasizes that humans, at least in most work situations, do not normally wander or run about aimlessly. Instead, the decisions to travel and the methods by which the travel is accomplished are goal-directed and ruled-based. The requirement for a person to TRAVEL TO specified-place must, therefore, be a fairly complex set of activities triggered by the need to go to a new place. One may assume that humans (at least, most adult humans) have learned and have stored general strategy procedures and algorithms by which they make intelligent decisions regarding how they will travel to the new location.

E-4.2 DETERMINE WAY TO Specified-place

When a person's general TRAVEL TO ... procedure is invoked (for whatever purpose), the "WAY" for getting from one's current location to the new specified place must first be determined. Thus, if the TRAVEL TO ... procedure is invoked, then DETERMINE WAY TO ... must be a very early mental activity that precedes the actual physical activities involved in moving from one place to another. It is likely that when a person has had to repeatedly travel from one specific place to another specific place, the optimal WAY will have already been learned, and can be retrieved from long-term memory. On the other hand, it is also likely that a person will also have learned general procedures for finding a WAY to unique places to which he has not been in the past. The generic problem-solving part of the DETERMINE WAY TO ... specified-place procedure undoubtedly includes various strategies that may result in heading in the general direction, looking for signs, or asking for directions.

E-4.2.1 The Concept of Multiple WAY-segments

The complete WAY to a new location may be composed of several WAY-segments, each of which must have both a PATH (e.g., open space, corridor, walkway, hallway, crawlspace, stairway, etc.) having a beginning and ending X, Y, and Z locus and a MOTION-TYPE (e.g., walk, run, crawl, etc.) to be used to traverse that space. Normally, a WAY segment will be a relatively straight or continuous path that leads to the next WAY-segment. Considered in this fashion, a body TURN while traveling can be thought of as simply the additional motion sometimes invoked by the need to transition (without stopping) from one WAY-segment to the next while engaged in TRAVEL TO ... Other changes in motions (e.g., starting, speeding up, slowing, or even stopping) may at times, be necessary to transition to the appropriate MOTION-TYPE for the next segment or to determine the next WAY-segment in case the current WAY-segment is blocked or the next WAY-segment must be determined.

E-4.3 TRAVERSE Specified-WAY-segment

When the WAY or WAYs to a new place have been determined (or, at least, when the initial WAY-segment(s) have been determined), execution of the required physical actions for traversing each successive specified WAY-segment can commence.

E-4.4 TRANSITION TO NEXT WAY-segment

As the end of one WAY-segment is approached, the person can commence transitioning to the next WAY-segment. If there are no remaining specified WAY-segments, and the specified-place has been achieved, then the TRAVEL TO ... procedure can be ended. If the specified-place has not been achieved, then the person will have to return to the DETERMINE WAY TO ... specified-place procedure to decide and specify the next WAY-segment to attempt.

E-4.5 SEARCH for OBSTACLES

During execution of traveling (i.e., following the planned WAY-segments), obstacles may be encountered. Because of this, physical travelling activity must usually be accompanied by perceptual/cognitive activity to SEARCH FOR obstacles. Here, we see an example of two different activities being prosecuted at the same time. Obstacles can be stationary or moving objects of various kinds, types of surfaces (e.g., slippery, burning, hot, etc.), or the lack of a sufficiently sturdy surface (e.g., a hole in a floor) that could interfere with or stop the normal execution of the planned travel.

E-4.6 OVERCOME Specified-obstacle

Once a potential obstacle has been recognized by the SEARCH FOR OBSTACLES procedure, other mental activities would be invoked to find a WAY to OVERCOME the specified-obstacle. Again, it can be anticipated that persons will have acquired general procedures for this purpose. Multiple strategies can be assumed to be present in this general procedure. Selection of a particular strategy that would be applicable probably depends primarily of the types of obstacle(s) found.

E-4.6.1 OPEN, AVOID, and REMOVE Specified-obstacle

When, for example, the obstacle is recognized as something like a closed door or hatch, the person will probably invoke a learned, general procedure containing strategies and action necessary to OPEN specified-obstacle. This procedure, in turn, may invoke various procedures to MANIPULATE specified-objects (e.g. turn a handle, pull/push handle, etc.). If the obstacle is a one that can be gone around, gone over, or gone under, the person may invoke a general procedure to AVOID a specified-obstacle. This procedure may consist of subdividing the current WAY-segment into yet smaller WAY-segments that provide an alternative path over, under, or around the obstacle. If an obstacle cannot be opened, or avoided, it may be the type of obstacle that can be removed to a new location. The REMOVE specified-obstacle procedure must first decide where the obstacle is to be placed. Then, a general procedure to MOVE specified-object TO specified-place can be invoked.

E-4.6.2 Resuming the Travel

Assuming that the person can determine a means to overcome the obstacle, the TRAVEL TO ... procedure can be resumed following the necessary actions required by the decision on how it is to be overcome. If, however, the obstacle cannot be overcome, the person may have to either determine a new path to the original desired place, or, possibly, the person will choose to wait for the obstacle to move or be moved by other forces.

From the above discussion, it may be seen that travel to a new place (regardless of whether it is accomplished by walking or by other methods) is a potentially complex set of activities. The mental decisions may, or may not, dominate the act of travel. In summary, several resident, general purpose procedures are probably required to account for the travel to a new place. They are:

```
TRAVEL TO ... specified-place.
DETERMINE WAY TO ... specified-place.
    TRAVERSE specified-WAY-segment.
        TRANSITION TO NEXT WAY-SEGMENT.
    SEARCH FOR OBSTACLES.
        OVERCOME specified-obstacle.
                OPEN specified-obstacle.
                        (MANIPULATE specified-object(s).)
```

body position in which the trunk and torso remain stationary for even short periods of time (regardless of whether one or more of the limbs or head and neck may be moving) can be referred to as a stance. For the most part, workers assume various stances in order to be in favorable positions to accomplish their tasks. Thus, stance changes are goal-satisfying movements where the goal is to get the job done. Every stance tends to lead to fatigue of one or more muscle groups Because of this, a worker may also, from time to time, stretch or shift to a somewhat different stance for no outwardly apparent reason. But even these stance changes are goal-satisfying movements in that they relieve muscle fatigue.

Decisions to bend forward, twist the torso, stoop, kneel, etc. are usually invoked to make it easier to access (look at or examine) something or to make it easier to reach and/or manipulate something. Thus, information about the particular task at hand, the actual physical placement of an object of interest in that task, and the current stance of the human should be sufficient to determine if a stance change is desirable and should be invoked. General rules (for a resident CONSIDER BODY-STANCE CHANGES procedure) may be written on which a simulated human can derive intelligent decisions as to the desirability to change stance. Because of this, it is unnecessary when describing a task or job to describe in detail when stance changes will occur.

E-4.8 CHANGE STANCE FROM ... TO ...

When a stance change is deemed to be desirable, accurate times to assess for transitioning from one stance to another should be able to be computed from knowing the beginning and ending stances, the load (if any) that is being carried at that time, the need for speed, etc.

E-4.9 Summary Comments on the Goal-Satisfying Approach

As was seen in Section E-3, a significant portion of the basic elemental motions of T&M systems were devoted to describing whole-body relocations, turning, and stance-changing actions. This section has shown that all of these actions are accomplished in order to meet other higher-level goals. That is to say, these types of actions are invoked, not because the job requires those actions, but because they are required by situations met while performing specified job tasks. In describing a person's job or tasks using the goal-satisfying approach, it is simply unnecessary to ever include an actual TRAVEL TO ... or CHANGE STANCE TO ... statement even though the job will definitely involve traveling to various places and stance changes. If, for example, the described job requires a certain object to be obtained from a particular location that is not fairly nearby, the TRAVEL TO ... procedure will automatically be invoked. If the person is currently sitting in a chair, and he must travel elsewhere, transition from a sitting to a standing position will be done prior to walking over to get the object. Again, this emphasizes the desirability of a goal-satisfying approach for a human operator simulation system. Such a system allows fairly simple descriptions of jobs to be further instantiated in greater detail by the system when certain actions become necessary in performing various tasks. That is, when the described job is actually simulated, the simulated external conditions occurring at appropriate times can be used to determine what specific actions or procedures are required to satisfy the job requirements. This, in turn, permits the calculation of task job times without having to tediously describe every possible elemental action or even the highly macro tasks such as walking. Such a system, however, will require a fairly detailed map of the spaces in which the simulated human performs his job.

E-5. THE APPROACH FOR DESCRIBING OBJECT HANDLING

As with whole-body motions, both similarities and differences existed among the four T&M systems for many of the basic upper limb motions/actions involved with obtaining, holding, releasing, positioning, aligning, and orienting objects. The concept of goal-satisfying strategy procedures in the human model can be equally well applied to this group of actions. Normally, people do not go around randomly picking up or discarding objects; such acts are done only because they serve some higher purpose or satisfy a particular goal.

E-5.1 The T&M Upper Limb Motions

When, for any reason, a person is required to use the upper limb parts to manipulate some object that is not already being held in his/her hand, the person will first have to OBTAIN (MSD) the object. Assuming the person is already very close to the object of interest, the person will have to first:

- use one or both upper-limb parts to reach for (REACH (MTM), TRANSPORT REACH [WFTS, MEWM]) the object.

At that point, the person might (to satisfy the goal):

- use upper-limb motions to manipulate the object (without grasping it) by pushing it in some direction, i.e., APPLY PRESSURE (MTM, WFTS) or EXERT FORCE (MSD).

However, if the person has to PICK UP (WFTS) the object in order to properly manipulate it, he/she will have to first:

- use the hand(s) and fingers to GRASP (MTM, WFTS, and MEWM) the object.

Having then gained positive control of the object, the person may now do a variety of things with it. For example, the person may:

- use the fingers and hand(s) to HOLD (WFTS, MEWM) it for a period of time and/or,
- use the fingers or hand(s) to position the object into a new position, e.g., POSITION (MTM), PRE-POSITION (WFTS, MEWM), FINGER SHIFT (MSD), POSITION (MEWM), ALIGN (MTM), ORIENT (MTM), ROTATE BY GRASP (MSD), TURN (MTM), CRANK (MTM), ROTATE BY CRANK (MSD) and/or,
- use arm movements (and possibly locomotion) to carry it to a new location (MOVE (MTM), TRANSPORT MOVE (WFTS), TRANSPORT LOADED (MEWM) or PLACE (MSD)).

Depending on the purpose or goal for which control of the object was obtained, the person may have to use some series of all three types of actions. Eventually, when physical control of the object is no longer needed, the person is able to:

- use the fingers and/or hand(s) to RELEASE (it) (MTM, WFTS), RELEASE LOAD (MEWM) or PLACE (it) ASIDE (WFTS).

While the four T&M systems do not specifically discuss relaxing the arms, hands, or fingers following the handling of objects, it is obvious that those actions may also take place from time to time.

E-5.2 "Generic" Procedures for Object Manipulation

Considering the types of actions/motions discussed, it can be seen that satisfying object handling goals are satisfied by a "generic" procedures or series of actions that include:

- TRAVEL TO it (if limbs not within reach of it)
- REACH TO/FOR it (if limbs not in contact with it)
- GRASP it (if object goal(s) requires that)
    - TWIST/TURN it (to new orientation) and/or
    - PRESS/PUSH it (to new orientation/location) and/or
    - HOLD it (at/in location/orientation) and/or
    - TRANSPORT it (to new location: may use TRAVEL TO)
    - PLACE it (at/on a desired surface) and/or
    - RELEASE it (if it is ready to be "ungrasped")
- RELAX limb parts (if they can now be relaxed).

In the above sequence of steps, it can be seen that the first three listed actions are simply "preparatory" actions to gain physical control of objects rather than object-manipulation goals per se. The object-manipulation goals are to either (a) change or (b) maintain an object's location and/or orientation.

E-5.3 Reactions to "Fatigue" and "Strain"

The "final" action to RELAX limb parts is usually done when the object-manipulation goals have been successfully accomplished and the body limb parts are not immediately needed in their current positions and their current positions are such that they would cause further "fatiguing" of some muscles. However, it is possible that some muscle groups will become fatigued prior to completely accomplishing all of the object manipulation goal(s). In that case, the normal procedure may have to be interrupted. For example, if a "heavy" object has been "grasped" and/or "held" for a long period of time, or is being "transported" over a long distance, it may be necessary for the human to rest some of the "fatigued" muscle groups by "placing" it on some surface and/or "releasing" the "grasp" of the object for a period of time. After sufficient rest (or "recovery" of those muscle groups), the object-manipulation goal(s) can be pursued again, but this may involve repeating some of the "preparatory" actions.

Interruption of some object-manipulation goals can also occur when "limits" of limb manipulation are reached prior to accomplishing the desired goals. For example, the wrist and hand can only turn a maximum number of degrees. If an object must be turned more than that limit, the object may have to be grasped, turned part of the desired way, and then released so that the hand and wrist can be readjusted to regrasp the object and continue turning it.

From the preceding discussions, it can be seen that, while object-manipulation goals are being pursued, the human must be obtaining and using information about the internal states of "fatigue" and "strain" on muscle groups and joints. Information on the particular body parts that require rest can be used to temporarily interrupt the object-manipulation goals and to decide how and when to obtain needed rest or time to recover. It is likely that human does not need to consciously "monitor" those states, but rather receives "alerting" interrupts when physical limits are being approached. Thus, it is likely that additional procedures available to allow the human to

REACT TO physiological strain.
INTERRUPT CURRENT TASK(S) TO PERMIT RECOVERY.

E-5.4 Decision Making during Object Handling

Finally, it can also be seen that the specific preparatory actions that must be taken are (or can be) the results of conscious decision making. For example, an object is "grasped" only when some object needs to be grasped to carry out further actions with that object.

With the exception of "grasp" and "release" that could, presumably, be accomplished by a barefoot human using toes, the generic procedural steps listed for object-handling goals can apply equally well for LEG MOTIONS (MTM) and FOOT MOTIONS (MTM) when an object is to be manipulated to a new position or orientation by using lower-limb actions.

E-5.5 Multi-Object Actions

Most of the actions that had been listed in Section E.3 for upper limb actions related to multi-object tasks (e.g., USE (WFTS, MEWM), ENGAGE (MTM), ASSEMBLE (WFTS, MEWM), DISENGAGE (MTM), and DISASSEMBLE (WFTS, MEWM)) should not be considered as "basic" actions or motions. Indeed, the titles and definitions of these offer little or no insight into the particular motions required. ASSEMBLE and DISASSEMBLE represent "time extensive" procedures that normally require specific knowledge of the components of an assembly, the specific tools, equipment, or materials needed for various components, and the sequence with which the components must be put together or taken apart. USE also normally represents a "time extensive" procedure that requires detailed knowledge on how to use specific tools, materials, or equipment for various purposes. As currently defined, ENGAGE deals with the motions necessary to "insert" or "join" one component into/to another component. DISENGAGE deals with separating an inserted or joined part from the part into which it was previously inserted or joined to. Obviously, specific information (usually learned in the past) is needed about the components to be able to properly join or take apart two components.

The preceding discussion makes it very obvious that the "multi-object" actions are not "basic" motions. Most of the actions in the previously described "generic" object-handling procedure would be required to assemble or disassemble components or to use tools. Thus, assembly and disassembly of equipment also represent higher level procedures that cause invoking of actions such as reach, grasp, twist, etc. The specific knowledge required to assemble, disassemble, or use equipment and tools ensures that "cognitive" processes are intimately involved in executing these higher level procedures. An important consideration that needs further investigation is how the structure of assemblies could be described so that general resident procedures could be invoked for either partial or complete disassembling or assembling of them could be accomplished.

E-6. THE APPROACH FOR DESCRIBING COGNITIVE ACTS

Cognitive acts or processes deal with a host of "mental" activities such as making decisions, drawing inferences, determining relationships, recalling facts, making plans, solving problems, interpreting information, mental calculations, and so forth. These acts, like others that humans do, are accomplished to satisfy various goals. The various T&M systems reviewed in Section E-3 had, for the most part, greatly neglected these cognitive acts. Only three categories of cognitive acts had even been mentioned:

PLAN -- deciding how to proceed with work

SELECT -- choosing one object out of a group, and

COMPUTE -- add, subtract, multiply, or divide.

Planning and selecting are obviously not "basic" mental acts even though they obviously involve mental activities. One of the problems associated with discussing cognitive acts is that they cannot be directly observed. Recently, the theory of underlying internal processes (UIPs) (Wherry, Jr., 1986) formulated a method for identifying the number and nature of the UIPs required for various kinds of tasks. Even here, though, one must assume that, in general, the approach used by all subjects in accomplishing variations of the task of interest will be similar. To the extent that some subjects use markedly different strategies in, say, planning some work, the UIP methodology may not be able to identify the UIPs being invoked.

It is obvious that computing is also not a basic mental act. For example, adding requires different processes than dividing. Even adding two-digit numbers together may require different processes that adding single-digit numbers. In Section E-6.1 the concept of UIPs and the need to model such basic processes is discussed briefly. In Section E-6.2 "learned" procedures which are assumed to be the basis for most complex mental acts are discussed. In Section E-6.3 the concept of "assertions" is developed which is a useful way to consider information received from external sources, retrieved from memory, or "inferred" from other available information. In Section E-6.4 decision making, how it is related to the manipulations of assertions, and the form that statements in procedures may take to accomplish complex tasks that require decision making are all considered. In Section E-6.5 the concept of procedures for mental calculations and computations are briefly discussed.

E-6.1 The Concept of UIPs and their MicroModels

The goal of some mental activities may be quite "simple" and involve a single type of mental "act." For example, to RECALL a "fact" (i.e., retrieve information from memory) may-involve a single type of cognitive activity. We may not fully understand the structure of human memory or the mechanisms by which that activity takes place, but we know that, sometimes, when we need to recall information, we can do it. How the act is accomplished is, to some extent, unimportant to us. We may not be able to describe what happened internally; all we know is that we needed that information and, shortly thereafter, we had it. Since it happened, we can assume that some "elemental," goal-satisfying, underlying internal process (UIP) is available to us or we would not be able to retrieve information from memory. We, somehow, "invoke" the UIP for RECALL, and the "fact" pops into our mind (or we realize that we can't recall the desired fact). We don't know exactly how that happens, but we would like to model the performance of that UIP. That is, we would like to construct a "*micromodel*" for RECALL that will tell us, whenever it is invoked, how long the RECALL process took, whether it was successful, and if so, the accuracy of the recalled information.

In the same way, we would like to have separate micromodels for all of the "basic" mental acts. UIPs are not limited to "cognitive" acts, but include sensory/perceptual acts as well as motor and speech acts. One of the reasons for establishing the micromodels for each identified UIP is to provide a method for handling individual differences among persons. That is, the times and accuracies with which individuals perform the *basic underlying* acts can effect their performance on many different tasks. The incorporation into a Human Simulation of UIP micromodels tha' include parameters for individual skills for those basic acts represents a reasonable strategy for accounting for individual differences in the performance of a wide variety of tasks.

E-6.2 The Concept of "PROCEDURES"

UIPs represent simple and basic acts. Mental activities may be "complex" in that they involve more than one or even a relatively large number of independent, different mental UIPs. That is, to satisfy the goal of a complex mental activity, the person must "proceed" through various "steps." We refer to those complex mental activities as a "procedure." We assume that the steps in one of these mental procedures have been learned and, consequently, reside in long-term memory.

The applicability of learned procedures can vary from extremely specific to extremely general. An example of a highly specific procedure might be one that permits us to use a certain type of equipment. A highly general procedure might be one that helps us to solve a general type of problem. For example, the procedures to "TRAVEL TO ... or DETERMINE WAY TO ... a specified place" (discussed in section E-4.6.2) are general procedures. Although not a requirement, a general problem-solving procedure will usually contain alternative approaches to accomplish the goal for which that procedure is invoked. A simple procedure will normally have a single approach (i.e., usually the simplest) by which the goal it is invoked for can be accomplished.

E-6.3 The Concept of an "ASSERTION"

"Assertions" play a central role in mental or cognitive acts and procedures. An assertion specifies the relationship (or lack thereof) between two (or more) things ("named-x" and "named-y"). The "specified-relationship" is interposed between the two named-things. The form of an assertion is, thus,

Named-x Specified-relationship Named-y .

E-6.3.1 "Facts" and "Definitions" as Assertions

The following four statements are all assertions:

"Joe is married (to Sue)."

"Joe is a male."

"Joe is taller than Sue."

"A married male is referred to as a 'husband'."

We might distinguish among the above assertions by saying that the first three appear to be stipulated "facts" while the last one is what we might call a "definition."

E-6.3.2 Deriving "Inferences" from Assertions

From the first two "facts", we "know" that "Joe is married" and also that "Joe is a male." Thus we know that "Joe is a married male." We also know (from the fourth assertion) that "a married male is called a 'husband'," so we may infer that Joe can be referred to as a husband. Our conclusion that: "Joe is referred to as a husband", is also of the form of an assertion, but it is an "inferred assertion" Thus, there can be many different types of assertions (e.g., facts, definitions, inferences, conclusions, etc.).

E-6.3.3 Deriving "Implications" from Assertions

Some specified relationships like "is married to" have the quality of "symmetry." That is if "A is married to B" then it is also true that "B is married to A." Other relationships do not possess symmetry. For example, the relationship "is taller than" does not have "symmetry," but it does have a "complementary" assertion. That is if "A is taller than B" then "B is shorter than A." The conclusion that "Sue is shorter than Joe" is based on a fact (Joe is taller than Sue) and the "knowledge" that the complement of "taller" is "shorter."

What is actually happening when we conclude that "Sue is shorter than Joe," is manipulating a series of word definitions. For example, one of the "attributes" of a person is "height" (or "vertical size"). Let us define some concepts, words, and conventions as follows:

"named-state"+"ER" MEANS "MORE named-state."
"TALL" IS OPPOSITE OF "SHORT."
"LESS" IS OPPOSITE OF "MORE."

The statement that "Joe is taller than Sue" can, thus, be translated to read, "Joe IS (MORE TALL) THAN Sue." The complement of any "IS MORE ... THAN" relationship would be "IS LESS ... THAN." Thus, it follows that:

"Sue IS (OPPOSITE OF) (MORE TALL) THAN Joe."

Now in the above statement we could substitute the opposite "MORE" or the opposite of "TALL" either would be proper. Thus both of the below statements are true:

"Sue IS LESS TALL THAN Joe," or
"Sue IS MORE SHORT THAN Joe."

But another way we could say "MORE SHORT" would be "SHORTER " thus, we could also say

"Sue IS SHORTER THAN Joe."

Normally, we would not think of this derived assertion as being an inference; we would think of it as being an "implication" that can be drawn from the earlier statement without any possible fear of being contradicted.

E-6.3.4 Representing "Goals" as Assertions

Finally, commands to ensure that something takes place can be represented as "desired goals." That is, the statements, "Move object-x to place-y," and "Manipulate control-x to state-y," can be represented as "goal" assertions (i.e., "assertions to be realized"):

"Object-x IS AT place-y." and
"Control-x IS SET EQUAL TO state-y."

Thus, even goals can be thought of and formulated as assertions. A "goal" is "realized" when the "goal assertion" can be said to be true.

E-6.3.5 The "Appropriateness" of the "Assertion Approach"

It should be noted that the above form for "goal" assertions cannot be distinguished from "facts" that the human might have been previously informed of, "sensed information" that the human may have perceived on his own, or "inferred information" that the human may have deduced from other available information. The concept that a large variety of different kinds of information can be represented and formulated as assertions permits the manipulation of previously learned facts, recently sensed or perceived information, inferences, implications, and stated goals in the same general ways. Such a capacity to combine factual information with inferential information or to combine previously learned information with recently acquired information corresponds to what we know about the capabilities of humans thought processes. The use of the "assertion" format for representing or depicting many kinds of information appears to be an appropriate and extremely robust approach for simulating human thinking.

E-6.3.6 Keeping Track of the "Sources" of Assertions

All information, however, may not be equally "valid," and its "validity" as well as its "probability" and "accuracy" of recall may well change across time. Thus, if assertions are to be manipulated for various "cognitive" acts, there is also a requirement for keeping track of the "source" of any assertion. As a minimum, "source information" should include "how" and "when" the assertion was originally provided. With that additional information, it would be possible to evaluate such things as:

- original "confidence" in the information
- likelihood that the assertion is no longer true

- probability that the human can recall it
- probable accuracy of recall, and
- present "confidence" in recalled information.

The concept of having "information" about "information" is also an extremely powerful concept, that permits the introduction of techniques for handling information processing under conditions of "uncertainty."

E-6.4 Decision Making

We tend to think of decision making as being involved for a host of related activities that run the gamut from simple to complex decision making. Some of these activities might include:

- select a particular object from a group of objects
- classify a target as a "hostile"
- decide when to "pull back on the stick"
- plan the steps in a project
- choose a particular course of action
- evaluate progress being made
- ... .

The action to "select object-x (from a specified-group of objects)" is, however, the result or consequences, of having made a decision about object-x and, perhaps, the other objects in the group. The statement tells us nothing about "basis" for the decision. A "decision" must involve the use or manipulation of some specific information that permits the narrowing down of the options available. The basis for deciding to "select object-x" may have been because it was the "handiest," or the "largest," or the "least expensive." The complexity or difficulty of making a decision is not specified by the resultant action, but, instead, by the consideration of how many options are available and the "criteria" used for determining which options meet those criteria. Suppose, for example, there are only two objects and one desires to select the closer of the two. The statement, "Select the closest object (to you) out of the specified-group" at least reveals the criterion ("closeness") to be used in making the decision. Note that the same information can be provided by the statement "If an object in the specified-group is closest (to you), then select it." What is interesting about this statement is that the phrase "an object in the specified-group is closest (to your)" is in the form of an assertion. That is the form of the decision statement is now:

IF assertion(s) THEN action(s)-to-take.

It is useful to think of the "IF" as a "conditional" word. That is, the use of "IF" makes the action(s)-to-take conditional on the assertion(s) being determined to be true. However, other words (e.g., "WHEN," "WHENEVER," "WHILE," "UNTIL," etc.) can be used in place of "IF." The "IF" is the form of the "conditional" for a decision statement where "testing" the validity of the assertion(s) is to be done one time only before proceeding to the next statement. Other "conditionals" indicate the need to continue "testing" the assertion(s) over a period of time. Multiple assertions can be "strung" together in a single statement by separating them with logical "ANDs" and "ORs" to form a more complex assertion. Finally, it should be pointed out that "action(s)-to-take" are, themselves, "goal assertions." This points up the "general form" for a decision statement is actually:

CONDITIONAL assertion(s) THEN assertion(s) .

This general form of the IF-THEN statement is extremely robust one that is applicable for many different kinds of decisions.

If one assumes that not only assertions, but also conditional statements can be committed to memory, then one can learn (and subsequently retrieve) general rules that can be applied to many different situations. A few examples of learned, basic "mathematical" rules would be:

"IF $B = A +$ a positive value then $B > A$"

"IF $B = A -$ a positive value then $B < A$"

"IF $A > B$ and $B > C$ then $A > C$"

"IF $A < B$ and $B < C$ then $A < C$"

"IF $A = B$ and $B = C$ then $A = C$"

"IF $A = B$ then $A + C = B + C$"

"IF $A = B$ Then $A * C = B * C$"

.... .

E-6.5 Mental Calculations and Computations

Adding, subtracting, multiplying, and dividing are obviously learned procedures. The use of paper and pencil during the use of any of those procedures may permit less reliance on short-term memory for intermediate results, thus improving accuracy for some persons. Both mental addition and subtraction and mental multiplication and division usually make use of "table lookups." Addition and subtraction tasks utilize a "sum" table while the latter two procedures use a table of the products of zero through nine multiplied by zero through nine. For some persons, parts of these tables may be "fuzzy" and lead to certain predictable errors and longer times when accomplishing some given arithmetic tasks. The table of learned "products" looks like:

						A					
		0	1	2	3	4	5	6	7	8	9
	0	0	0	0	0	0	0	0	0	0	0
	1	0	1	2	3	4	5	6	7	8	9
	2	0	2	4	6	8	10	12	14	16	18
B	3	0	3	6	9	12	15	18	21	24	27
	4	0	4	8	12	16	20	24	28	32	36
	...										
	9	0	9	18	27	36	45	54	63	72	81

The "products" (of A times B) can, of course, be considered as a table of assertions. The time and accuracy of recall of these assertions (as needed) in the arithmetic tasks is thus one of the major determiners of the time and accuracy of performing the particular arithmetic tasks. Access to recall of a cell in the table seems to be direct and almost immediate as we would expect from any well learned "fact". From this standpoint, the time to perform a multiplication task appears to be predominantly determined by the number of recall (and "carrying") operations required by the task. The individual must know the "procedure" for multiplying, but the time and accuracy of the result may be mostly determined by the UIP for RECALL.

Other "mental" calculations require the knowledge of equations or other rules. Here again, though, we see that a rule or an equation (e.g., for the length of the hypotenuse of a right triangle) is simply a learned assertion.

It may well be that the number of "cognitive" UIPs are relatively few. Certainly, RECALL ... and IF-THEN ... (i.e., determine the "truth" of an assertion) appear to be fairly elemental cognitive acts. It is also likely that we often attempt to STORE information for subsequent recall by "rehearsing" the information. Thus, STORE ... may also be considered to be an elemental cognitive act. Most of the more complex cognitive acts appear to be applying a rule or group of rules to some assertion or group of assertions. We have already seen that drawing of inferences and implications involves the evaluation of more than one assertion. However, that type of process would appear to involve (and may be able to be handled by) sequential retrieval of rules and other assertions and simple decision making. It is likely that all complex cognitive acts are procedu^re- or rule-based that simply involve the obtaining of information (through RECALL or perceptual processes) as needed and simple decision making. One of the issues this raises is the nature and number of "resident" strategy procedures that need to be included in the human model.

A recent factor analysis (Wherry, Jr., 1987b) of the ASVAB and "Project A" tests on 4733 soldiers revealed two independent dimensions for recall of knowledge from long-term memory. One of these dealt with "Technical Knowledge" and the other appeared to deal with "Vocabulary Knowledge." Separate factors were also identified for "Decision Making Speed" and "Logical Reasoning Speed." The fact that two independent factors emerged from the analysis suggests that drawing logical inferences from assertions may well be a different elemental cognitive process from simple decision making.

E-7. THE APPROACH FOR DESCRIBING VISUAL/PERCEPTUAL ACTS

E-7.1 "Basic" Visual Actions

As can be seen in Section E-3, T&M systems listed many different "elemental" motions that relate to visual acts. However, these range from simple "eye movements" to complex acts such as "reading" and "searching for an object." While there is no argument that these involve visual processes, the extent to which they should be considered "elemental" is an important issue for the development of HOS IV.

E-7.1.1 Determining the Next-Place-to-Fixate

The primary function of the eyes is to receive information available in the external visual world. Because of the finer "grain" available in the relatively small, central foveal area, more detailed visual information can be received there than from the areas in the peripheral visual field. Some information available in the periphery, however, can be assessed and evaluated to aid in some decisions regarding subsequent areas in the visual field to bring into the foveal area. Recalled information on the probable location of objects can also be used to determine necessary eye movements to fixate a desired object. Eye movements can also be made to follow a predetermined sequence or pattern (e.g., left-to-right, top-to-bottom, etc.). Thus, one of the "basic" actions related to vision is to determine the "next" place to fixate, and there are at least four "processes" available for accomplishing that goal:

- DETERMINE next-place-to-fixate USING:
  - characteristics of objects sought
  - places of interest in the peripheral field
  - the recalled place of an object, and/or
  - the next place in a predetermined pattern.

E-7.1.2 Moving the Eyes to the Next-Place-to-Fixate

Movements of the eyes are accomplished using sets of muscles that permit the eyes to be shifted (EYE TRAVEL (MTM) and EYE SHIFT (WFTS)) in azimuth and elevation and to be refocused (EYE FOCUS (MTM and WFTS)) on objects or points of interest at various distances.

Because different muscles are involved in eye shifts and eye focusing, both actions can apparently be going on at the same time as the eye changes from one fixation point to another. Because of this, both actions, as necessary, can be subsumed under:

MOVE EYES TO next-place-to-fixate.

E-7.1.3 Absorbing the Information in the Foveal Area

Having fixated a given object or place of interest, the detailed information now available in the foveal area can then be perceived. The speed and accuracy of perceiving particular visual information in the foveal area is undoubtedly dependent on the type of information available there and the type of information being sought. Regardless of what parameters control the speed and accuracy of absorbing information in the foveal area, it represents a different visual/perceptual process.

While visually absorbing/perceiving of information may appear to be an elemental visual act, recent evidence (Wherry, Jr., 1987a) indicates that separate (i.e., independent) processes are required for absorbing color, shape, size, and orientation of objects.

Yet other independent processes may also be involved (Wherry, Jr., 1987b) in reading and comprehending words and other alphanumeric strings. Thus, depending on the nature of the visual stimuli, there may be a variety of different elemental visual/perceptual processes that can be invoked by:

- ABSORB fixated-information USING:
  - color
  - shape
  - size
  - orientation
  - alphanumerics
    - word strings
    - numeric strings.

E-7.2 A Generic Procedure for Obtaining Desired Information

While the preceding discussed kinds of visual actions (determining where to look, fixating the eyes at a new position, and perceiving visual information of various types) are basic, they are normally accomplished in sequence because the human needs to find and obtain certain desired "displayed" information, a desired object, or the state of some object. Obtaining information on the location of a particular object or an objects of a certain kind can be thought of and expressed as finding that (kind of) object. Similarly, finding other specific information about an object can be thought of as determining/estimating the condition or state of some object. Thus, the three basic kinds of visual actions are "triggered" or "invoked" in sequence to satisfy the human's higher-level goal that can be stated as:

OBTAIN desired-information.

Many of the visual/perceptual actions mentioned in the four T&M systems actually fall into this higher-level goal category (e.g., SEARCH (MEWM), FIND (MEWM), INSPECT(ION) (WFTS, MEWM, and MSD), READING (MTM), ACTION READ (WFTS), CONCEPT READ (WFTS), VISUAL CHECKING (MSD), VISUAL COMPARISON (MSD), and REACT (WFTS)).

It must be assumed that "OBTAIN desired-information" is actually a fairly complex learned generic procedure involving various memory retrieval and decision making actions. The following sections discuss a generic strategy for obtaining any desired information.

E-7.2.1 Recall of Sources of Desired-Information

Up to now, the only discussed source of the desired information has been through the use of vision. However, desired information may be recalled from memory or acquired through a variety of senses. One type of information that can normally be recalled from memory includes the type(s) of source(s) that could be used to obtain the desired-information. Thus, the first steps in obtaining the desired-information is likely to be:

RECALL source(s) of desired-information.

E-7.2.2 Recall of Desired-Information

One of the sources might be short-term memory (STM). Thus, the next step in the OBTAIN desired-information procedure might be:

IF source IS "STM" THEN RECALL desired-information.

It is possible that the desired information may be able to be satisfactorily recalled in which case the desired information will have been obtained and the generic procedure for obtaining information can be ended. That is, the next step might be:

IF desired-information IS RECALLED THEN QUIT.

However, if the information cannot be successfully recalled or if there is reason to believe that the recalled information may no longer be accurate and one of the recalled sources of the desired information was "visual," then one might expect a separate strategy procedure to be invoked that "handles" the acquiring of visual information. Thus, the next step in the "OBTAIN desired-information" procedure would be:

IF source IS vision THEN ABSORB desired-information.

E-7.2.3 Visually Absorbing Desired-Information

The first step in the "Visually ABSORBing" procedure may well be:

IF eyes ARE NOT FIXATED ON source THEN FIXATE source.

Similarly the first step in the FIXATE procedure may be:

IF location OF source IS NOT KNOWN THEN RECALL it.

E-7.2.4 Searching for Objects and Sources of Information

If the location of the source is successfully recalled then the eye-movement microprocess is triggered and successful fixation of it will follow. However, it is possible that the source of the information cannot be recalled and the next step might be:

IF source location IS NOT RECALLED THEN SEARCH FOR it.

At this point that the "DETERMINE next-place-to-fixate (NPTF)" will be triggered. Assuming that a "next-place-to-fixate" is forthcoming, and that place is "within-sight" then the MOVE EYES TO NPTF" action will be taken. (If the NPTF is not believed to be within sight, the human may either abandon trying to look at the NPTF or may decide to "TRAVEL TO a place where the NPTF may be seen.") If in trying to look at the NPTF, the person's vision is somehow "blocked" from doing that, then he may return to the previous step to decide if travelling to a new vantage point is desirable or not. Assuming that the person does finally move the eyes to the NPTF, the information available at that point (or in that area) may be perceived. If the information found there is what was desired, then the original triggering goal will have been satisfied. If not, procedural control may revert to the "DETERMINE next-place-to-fixate" action.

It can be seen that sooner or later, the source of the desired information should be found so that it can be fixated and, ultimately, the desired information can be absorbed.

The concept of "SEARCH (MEWM)" may certainly be like that described above. "INSPECT" (WFTS, MEWM, and MSD) may also be accomplished by looking at a sequence of different objects or places for various reasons. "INSPECT," however, implies previously developed knowledge of the sequence of places that must be looked at. The definitions supplied for "READING (MTM)" and "CONCEPT READ (WFTS)" seem to imply the type of reading done when a passage of text is the information to be absorbed, although it may also apply to a table of numbers. In such cases, the pattern of scanning the material to be read is well established (or can be determined from peripheral cues as the reading takes place). "ACTION READ (WFTS)," "VISUAL CHECKING (MSD)," and "VISUAL COMPARISON (MSD)" are so ill defined that they may be applied to the perceiving of information from displays, text, tables, or symbolic displays. All seem to imply that the information perceived will have to be compared to other information, the source of which is not specified.

Recent studies (Wherry, Jr., 1987b) have demonstrated evidence of separate processes for:

- Visual Scanning and Recognition Speed and
- Eye-Movement Speed.

This indicates that the capability for peripherally searching for objects is probably independent from eye movement speed and should, as indicated above, be considered as separate microactions. A separate factor was also identified for ta-ks that involve Spatial or Figural Orientation. This suggests that, in some tasks, images may have to be "mentally" rotated to different orientations in order to satisfy task goals. This type of an act could be considered to be a visual or perceptual act or a cognitive one.

E-7.3 Summary of Visual/Perceptual Acts

In the above discussions, it has been seen that much of what is normally considered to be visual/perceptual activity requires both information recall (from both short-term and long-term memory) and decision making Convincing evidence exists that indicate a variety of different visual / perceptual processes are needed to account for individual differences in the performance of various visual/perceptual tasks. It also appears that the higher level, general strategy procedures for obtaining any desired information are likely to first attempt recall that information. If that fails, recall of an alternate source of the desired information may be attempted. If that too fails, the human will probably attempt to search for the needed information As with whole-body, object-manipulation, and cognitive actions, visual/perceptual acts can be modeled using both "elemental" processes and resident strategy procedures.

E-8. CONCLUSIONS AND RECOMMENDATIONS

E-8.1 CONCLUSIONS

Review of the T&M systems was useful in constructing a general approach for how HOS IV could deal with actions related to locomotion and object manipulation. The four T&M systems evidenced significant disagreement among themselves as to what constituted the "basic" or "elemental" actions and motions. By including consideration of the actions/motions of four different T&M systems, it may be concluded that all of the important actions relative to locomotion and object manipulation have probably been covered.

The separate discussions for whole-body, object handling, cognitive, and visual/perceptual actions have indicated that HOS IV micromodels can be developed that permit it to be used to realistically simulate jobs that require the human to move about a workstation and manipulate objects. During the development of HOS IV micromodels, careful attention must be given to distinguishing between basic or elemental human acts (i.e., simulated "microactions") and strategy modes that have been acquired by humans and represent learned knowledge. Many new words will have to be added to the simulation language to allow expression of the activities associated with locomotion and object handling. Successful development of HOS IV will be realized when it becomes:

- easy for the user to describe human tasks using the augmented language and
- the human model available to HOS IV is capable of producing realistic human performance of those described tasks.

E-8.2 RECOMMENDATIONS

This document has provided insights and initial guidance for the development of HOS IV micromodels capable of simulating human tasks that involve locomotion and object handling in addition to various cognitive and visual/perceptual activities. The approach advocated for that development includes the use of many different general-purpose strategy procedures in the Human Model of HOS IV. Included among the needed resident procedures for locomotion and object handling are:

TRAVEL TO ... specified-place
DETERMINE WAY TO ... specified-place

TRAVERSE specified-WAY-segment
TRANSITION TO NEXT WAY-segment
SEARCH FOR OBSTACLES
OVERCOME OBSTACLES
– OPEN specified-obstacle
– REMOVE specified-obstacle
– AVOID specified-obstacle
CONSIDER BODY STANCE CHANGES
CHANGE STANCE FROM ... TO ...
REACH TO/FOR specified-object
GRASP specified-object
– TWIST/TURN specified-object ...
– PRESS/PUSH specified-object ...
– HOLD specified-object
– TRANSPORT specified-object ...
– PLACE specified-object ...
RELEASE specified-object
RELAX specified-limb-parts
ASSEMBLE/DISASSEMBLE specified-parts
REACT TO physiological strain, and
INTERRUPT CURRENT TASK(S) TO PERMIT RECOVERY.

Additionally, this report discussed various other cognitive activities that will be required in the expanded HOS IV that are related to storing and recalling information, drawing inferences, making decisions, and planning. Many of these activities are also required for enabling the more cognitive portions of the previously mentioned general-purpose resident strategy procedures. Additional study of the most effective ways of storing, retrieving, and manipulating "assertions" in HOS IV is needed.

Finally, discussions were presented on various visual/perceptual activities required for HOS IV to be able to search for and locate objects and other information in the simulated human's workstation. Detailed evidence exists for the existence of individual differences in the capability for visual peripheral searching that is independent of both eye movement time and foveal perception of information. The addition of a resident peripheral search procedure and the differentiation of perceptions of various visual stimuli for HOS IV would greatly enhance the user's ability to describe various jobs to be simulated by the new HOS.

E-9. REFERENCES

Antis, W., Honeycutt, Jr., J. M. & Koch, E. N. (1968). The basic motions of MTM. The Maynard Foundation, Pittsburg.

Crossan, R M. & Nance, H W. (1962). Master Standard Data, the economic approach to work measurement. McGraw-Hill, New York

Karger, D. W. & Bayha, F. H. (1966). Engineered work measurement. Industrial Press, New York.

Maynard, H. B., Stegemerten, G.J., & Schwab, J. L. (1948). Methods-Time Measurement. McGraw-Hill. New York.

Morrow, R. L. (1957). Motion economy and work measurement. Ronald Press, New York.

Quick, J. H., Duncan, J. H., and Malcolm, J. A. (1962). Work Factor Time Standards. measurement of manual and mental work. McGraw-Hill, New York.

Wherry, Jr., R. J. (1986). Theoretical developments for identifying underlying processes; volume 1: The theory of underlying internal processes. NAMRL Special Report 86-4 and NADC Report 86 105-60.

Wherry, Jr., R. J. (1987a). Internal processes required for using displayed tactical information: study #2. Technical Report 2100.03-TR-01. Willow Grove, PA: Analytics, Inc.

Wherry, Jr., R. J. (1987b). Factor analysis of Army ASVAB and Project A data, Technical Report 3134-1, R. J. Wherry, Jr., Chalfont, PA.

APPENDIX F

FACTOR ANALYSIS OF ASVAB AND SUPPLEMENTARY TESTS OF PROJECT A

ROBERT J. WHERRY, JR., PH.D.

F-1. FACTOR ANALYSIS OF ASVAB AND SUPPLEMENTARY TESTS OF PROJECT A

F-1.1 Purpose of the Factor Analysis

The usual purpose of a factor analysis of any matrix of intercorrelations among various test scores is to identify the number and nature of the independent factors causing "good" or "poor" performance. The specific purpose for which this factor analytic study was undertaken was to identify individual differences that must be accounted for in a human model capable of reflecting an individual soldier's scores on the U.S. Army test battery currently undergoing development and validation (e.g., Wing, 1985) for its use in the MANPRINT Method Products. Hopefully, factors identified as required in the new selection battery would be equally important in performing actual Army tasks.

This report can be considered preliminary in that a) larger number of subjects would have been desirable, b) time did not permit an indepth examination of the variables for possible 'outliers' and c) no attempt has yet been made to determine the relevance of the factors identified with various Army field test scores for different Military Occupational

F-2.1 The Factor Analytic Procedures Used

The intercorrelations among scores on thirty-nine selected variables whose data had been collected by the Army on 4,733 soldiers were factor analyzed using the Principle Factors Analysis (PFA) procedure. Fifteen common factors were extracted by the PFA procedure. That is to say, the intercorrelations among the test variables can be accounted for by only fifteen independent dimensions or factors. However, because the factor extraction process is based strictly on mathematical techniques that extract as much variance as possible from the matrix for each succeeding factor, extracted factors are rarely, if ever, "meaningful" or "interpretable." To arrive at factors that are meaningful, the factors must first be rotated in space. One of the traditional rotation techniques is the Varimax procedure (Kaiser, 1959) that attempts to find a solution having what is known as "simple structure." Therefore, the fifteen extracted factors were submitted for Varimax rotation. Unfortunately, simple structure could not be found and Hierarchical Factor Analysis (HFA) using the Positive Manifold technique (Wherry, Jr., 1986) was then used to identify any "higher order" factors that were precluding simple structure. Four "general" factors (i.e., factors that tend to make variables load positively on all Varimax-rotated factors) were identified by the HFA procedure. The nineteen HFA factors (i.e., the four general factors and the fifteen residual specific factors) were finally rotated by a computerized graphical rotation technique. One specific factor subsequently "collapsed" (i.e., its loadings were able to be rotated to other factors) leaving a final total of eighteen orthogonal factors to be interpreted.

F-3.1 Variables Included in the Analysis

Variables included the subscale tests of the ASVAB (Armed Services Vocational Aptitude Battery), various measures of performance on several computerized tests, additional paper-and-pencil tests, and height and weight. The names of the thirty-nine variables, their means and their standard deviations are listed in Table 1. Explanations of the tests can be found in various ARI publications.

Those variables without asterisks are performance measures in which a higher score would represent "better" performance (e.g., accuracy, percent correct, number of items completed, etc.). Those variables with asterisks represent performance measures in which a higher score would indicate "poorer" performance (e.g., average or total time to complete the test or its items, average error made, total amount or number of errors, etc.). The factor loadings of asterisked variables have been "reflected" (i.e., multiplied by -1.) so that a "positive" loading on any variable indicates "desired" performance (i.e., both fast and/or accurate responses).

F-4.1 Identification of Factors

Each factor found represents an independent theoretical construct (e.g., an underlying stable individual difference in some internal process or strategy that causes performance on one variable to be related (positively or negatively) to performance on other variables). However, factors are rarely, if ever, directly measured by any single variable. Indeed, if a given factor influences soldiers' scores on less than two of the variables in the correlation matrix, it can never be identified by the factor analytic approach because factor analysis is only concerned with the analysis of variance that is "common" to two or more variables. The nature of a given factor can usually be identified by both the variables that have significant loadings on a factor as well as the

Table 1 -- Graphical Rotations for the Army ASVAB & Project "A" Data

Name of Army Test	ROTATED FACTOR LOADINGS																		Comm
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
ASVAB VE: Verbal	65	22	-02	01	76	-02	-04	01	00	01	01	-02	01	02	-03	01	00	01	1043
ASVAB WK: Word Knowledge	61	22	-04	00	66	01	-03	01	-04	00	01	02	01	-02	-03	00	01	-02	856
ASVAB PC: Paragraph Comprehension	53	18	04	00	49	02	02	01	07	01	01	-04	01	03	-01	-01	01	03	568
ASVAB GS: General Science	58	39	00	03	35	13	07	03	-02	04	01	05	02	-02	-01	01	01	00	639
ASVAB AS: Autoshop Information	33	61	-03	02	-02	34	-06	06	-03	-05	-02	00	-01	04	-03	07	01	-09	616
ASVAB EI: Electronic Information	42	54	-04	-03	03	33	02	-04	03	-01	04	07	01	01	02	00	-02	-01	588
ASVAB MC: Mechanical Comprehension	46	58	03	00	04	23	20	14	-02	07	02	01	01	-01	01	-01	-03	03	672
ASVAB AR: Arithmetic Reasoning	71	11	23	01	-04	02	33	01	01	05	00	-03	01	-02	-02	-02	02	-07	689
ASVAB MK: Mathematics Knowledge	71	04	20	04	-02	-04	41	-07	07	08	02	02	00	-03	06	00	01	08	737
ASVAB CS: Coding Speed	19	-21	34	01	07	01	02	06	53	-02	01	01	00	03	03	01	-01	21	526
ASVAB NO: Numerical Operations	23	-36	43	11	-06	-02	-03	-05	48	08	-02	00	01	-03	-05	-04	01	03	611
CORR: Reasoning Test	41	21	27	-04	06	-07	29	37	-02	01	-02	03	01	06	03	-02	02	02	521
CORR: Maze Test	10	35	36	12	04	02	10	51	13	05	01	-02	00	-05	-01	01	04	02	569
CORR: Assembling Objects	26	09	08	-07	02	-03	26	47	01	-06	01	05	-02	04	07	-04	-03	05	531
CORR: Object Rotation	12	33	28	13	-02	05	10	43	09	-02	-01	-02	03	-03	-02	04	05	00	432
CORR: Orientation Test	33	33	16	-02	-01	01	27	29	-02	03	-02	-02	02	02	04	-02	03	07	403
CORR: Map Test	47	31	24	-02	04	04	26	28	02	01	-03	03	-01	01	04	03	-01	01	526
• Tgt.Tracking 1: Avg. Log (Dist+1)	06	52	32	01	-04	-04	06	-07	-03	84	01	00	00	04	12	-03	-04	01	1111
• Tgt.Tracking 2: Avg Log (Dist+1)	07	52	32	03	-02	-01	05	-04	01	75	00	00	-02	01	13	-03	-04	02	962
• Cannon Shoot Avg. Abs. Time Error	05	34	31	05	02	02	-02	15	03	32	05	-01	04	-05	02	06	-02	-13	366
• Tgt. Shoot: Avg. Log (Dist+1)	-06	28	26	-22	03	01	01	00	01	44	37	03	-01	-02	03	02	02	00	525
• Tgt. Shoot: Avg. Time to Fire	09	26	13	32	-03	-03	01	03	04	35	-34	02	02	00	07	01	03	02	443
• PSA: Avg. Trimmed Decision Time	-06	04	09	74	13	-05	-06	18	11	02	26	17	-02	07	05	-02	-03	00	732
PSA: Avg. Hit Rate	19	-05	24	-62	03	00	-01	06	01	04	-24	03	-01	-03	02	-03	01	-05	534
• Tgt. ID: Avg. Trimmed Decision Time	07	35	19	58	14	01	-01	34	05	16	06	14	03	-02	01	04	-01	-05	684
Tgt. ID: Avg. Hit Rate	11	05	16	-50	06	00	04	11	01	-03	02	-07	00	06	01	02	-02	-01	307
• Avg. Trimmed Simple Reaction Time	-04	12	20	06	-01	00	05	03	01	01	01	01	65	13	04	01	01	04	511
• Avg. Trimmed Choice Reaction Time	07	-02	19	13	03	03	00	04	07	27	-02	53	52	-18	-08	-02	-02	01	727
Simple Reaction Avg. Hit Rate	13	06	03	-02	-02	01	03	01	02	-03	01	-09	16	41	04	-05	02	03	227
Choice Reaction Avg. Hit Rate	20	01	-01	12	-01	00	-01	06	00	05	00	03	-10	48	-02	01	-01	-03	295
Memory: Avg. Hit Rate	16	06	28	19	14	-05	00	21	-02	-05	-01	05	-03	12	10	02	-04	-07	238
• Memory: Avg. Trimmed Decision Time	00	04	19	29	-03	-01	02	06	00	04	03	52	06	-04	27	04	04	11	483
• Pooled Avg. Movement Time	11	23	33	12	-03	-03	03	01	01	-03	01	-07	02	04	40	01	01	11	367
• Number: Avg. Final Response Time	31	07	47	26	01	02	-02	04	00	-04	-04	24	-07	05	31	-03	22	-05	605
Number: Avg. Hit Rate	-37	06	-45	11	-04	-01	01	-08	01	-03	01	02	01	-03	-01	01	08	31	445
• Number: Avg. Initial Input Time	26	02	29	08	06	-02	-03	09	01	00	-02	-02	03	-01	-02	-03	54	01	466
• Number: Pooled Avg. Oper. Time	45	-08	28	28	-05	-02	07	-04	02	-03	03	03	03	01	-01	02	61	00	745
Height	02	04	01	05	-01	01	02	01	-02	04	01	02	00	-06	-01	70	-01	00	620
Weight	00	28	00	03	01	00	00	-01	-01	-06	01	-02	01	04	03	73	00	01	612

Note: a) Decimal place omitted b) variables with asterisks were reflected, c) loadings higher than .10 are significant.

Table 2 -- Names, Means, Standard Deviations, Highest and Lowest Scores for Army ASVAB & Project "A" Data (Number of Cases = 4733)

	Name of Army Tests and Subscales	Mean	Standard Deviation	Low Score	High Score
	ASVAB VE: Verbal.	50 667	6.5122	26	62
	ASVAB WK: Word Knowledge	50.357	6.8330	29	61
	ASVAB PC: Paragraph Comprehension	51.221	6.8019	20	62
	ASVAB GS: General Science.	51.206	8.0714	24	68
	ASVAB AS: Autoshop Information	54.098	8 4779	28	69
	ASVAB EI: Electronic Information	51.765	7.4932	23	70
	ASVAB MC: Mechanical Comprehension.	53.098	7.9096	25	70
	ASVAB AR: Arithmetic Reasoning	52.727	7.1311	34	66
	ASVAB MK: Mathematics Knowledge	50.743	7.2244	32	68
	ASVAB CS: Coding Speed	51.305	6.7108	26	72
	ASVAB NO: Numerical Operations	52.614	6.4132	26	62
	CORR: Reasoning Test	19.593	5.4253	0	30
	CORR: Maze Test	6.717	4.6801	1	24
	CORR: Assembling Objects	23.704	6.3999	0	32
	CORR: Object Rotation	63.200	18.9714	1	90
	CORR: Orientation Test	11.359	6.1330	0	24
	CORR: Map Test	8.078	5.5156	0	23
*	Tgt.Tracking 1: Avg. Log (Dist+1)	2.965	.4721	1.977	5.147
*	Tgt.Tracking 2: Avg. Log (Dist+1)	3.678	.5047	2.260	4.963
*	Cannon Shoot: Avg. Abs. Time Error	43.440	8.7810	21.360	106.285
*	Tgt. Shoot: Avg. Log (Dist+1)	2.171	.2400	1.627	4.915
*	Tgt. Shoot: Avg. Time to Fire	234.720	47.8851	64.167	435.724
*	PSA: Avg. Trimmed Decision Time	235.870	61.9100	29.263	431.466
	PSA: Avg. Hit Rate	.876	.0792	.500	1.000
*	Tgt. ID: Avg. Trimmed Decision Time	190.090	60.4776	52.643	456.727
	Tgt. ID: Avg. Hit Rate	.911	.0703	.500	1.000
*	Avg. Trimmed Simple Reaction Time	31.432	13.5206	19.000	291.111
*	Avg. Trimmed Choice Reaction Time	40.639	9.6446	20.667	250.799
	Simple Reaction Avg. Hit Rate	.985	.0426	.571	1.000
	Choice Reaction Avg. Hit Rate	.986	.0307	.552	1.000
	Memory: Avg. Hit Rate	87.116	23.8026	34.333	301.799
*	Memory: Avg. Trimmed Decision Time	.890	.0735	.500	1.000
*	Pooled Avg. Movement Time.	33.218	7.7548	14.561	79.342
*	Number: Avg. Final Response Time	157.800	41.4618	63.444	372.583
	Number: Avg. Hit Rate	99.100	.0872	99.000	99.500
*	Number: Avg. Initial Input Time	140.110	54.1477	51.154	829.408
*	Number: Pooled Avg. Oper. Time	230.010	78.4502	29.806	840.360
	Height	68 600	3.0432	58	78
	Weight	153.640	25.6770	93	265

Note: Asterisked Variables indicate high score = poor performance.

variables that do not load on that factor. The final eighteen graphically-rotated factors are tentatively named and interpreted in the subsections that follow. Table 2 shows the final factor loadings for the various test measures.

F-4.1.1 Reading Comprehension Speed

This widely-used, "general" process permits written verbal material to be rapidly read and understood. All of the ASVAB subtests loaded positively on this factor (as might be expected for written tests. most of which require reading of test items). All of the CORR "paper and pencil" tests also loaded positively on this factor. Of the computerized tests, several of the criterion scores for the Number test (which also requires reading of numbers) also loaded on this factor (although the negative loading (-.37) of Number: Average Hit Rate was surprising and possibly due to the lack of variance on this variable [See Table 1]). All of the remaining computerized tests that loaded on this factor were positive loadings for Average Hit Rates; e.g., Target ID (.11), Choice Reaction (.20), Simple Reaction (.13), PSA (.19), and Memory (.16). This may indicate that the instructions for these tests were better understood by those possessing good reading comprehension skills.

F-4.1.2 Practical Experience and Skill

This factor is particularly interesting in that it probably represents "general" capabilities and skills for spatial/technical tasks that have been acquired through practice and experience. It is independent of the factor six (Technical Knowledge) which may represent additional technical "book" knowledge that has been acquired but may not, as yet, have been applied in real situations, and may not be particularly useful in performing well on any of the CORR or computerized tests.

The four ASVAB subtests that contained test items of a technical nature (Autoshop Information (.61), Mechanical Comprehension (.58), Electronic Information (.54), and General Science (.39)) loaded quite highly on this factor, and only ASVAB Mathematics Knowledge failed to show a significant loading on this factor. It is assumed that many of the test items on most of the ASVAB subtests require knowledge that could have been acquired through experience on practical tasks. The fact that the reflected Target Tracking Average Error scores also had a high loadings (i.e., both .52) on this factor was what first suggested the probable importance of practical experience for this factor. All six of the CORR tests also had significant positive loadings on this factor ranging from .21 to .35. This could also be explained as the result of having personally experienced situations similar to those presented in these six tests. The negative loadings on the ASVAB Numerical Operations (-.36) and Coding Speed (-.21) may indicate a greater neglect for "clerical" type tasks in favor of those requiring dynamic information processing and action on the part of the soldiers.

Opportunity for gaining practical experience is also suggested by the fact that both Height (.34) and Weight (.28) also had significant loadings on this factor. Two reasonable explanations of opportunity to acquire experience concern the age and sex of soldiers. With regard to age, younger soldiers can be expected to be shorter (and consequently weigh less) than older soldiers. But older soldiers will have had more opportunity for gaining practical experience. With regard to the sex of soldiers, males can certainly be expected to be taller and heavier than females, and it is likely that they will have experienced more technical situations that are like those presented in the four ASVAB subtests mentioned earlier. The fact that several criterion scores for some of the computerized tests in addition to Tracking (e.g., Cannon Shoot, Target ID, and Target Shoot) also loaded on this factor again reinforces the interpretation that the skill represented by this factor may have derived from practice on similar tasks.

F-4.1.3 Visual Scanning and Recognition Speed

Visual scanning and recognizing would be an important "general" process in tasks requiring persons to rapidly scan a visual field and to find, fixate and correctly interpret displayed material containing groups of alphanumerics, symbols, or objects. From Table 1 it can be seen that all of the CORR tests and most of the computerized tests (when properly reflected) had significant positive loadings on this factor. Again the relative loading of Number average Hit Rate is probably attributable to the lack of variance on this variable. The fact that the ASVAB tests for Numerical Operations (.43) and Coding Speed (.34) lend support this for this explanation since both require a high degree of visual scanning of the information printed on those forms. Similarly, the ASVAB Arithmetic Reasoning (.23) and ASVAB Mathematical Knowledge (.20) require scanning of numbers and symbols other than words to attain high test scores.

F-4.1.4 Speed/Accuracy Tradeoff Strategy

A "general" speed versus accuracy tradeoff strategy for an individual could be expected to result in a consistency with which that person proceeds through many kinds of sequences of operations in which either more time will be spent to gain a higher accuracy or accuracy may be sacrificed to obtain faster response times. Table 1 shows that several computerized test criteria (e.g., PSA (.74 and -.62), Target ID (.58 and -.50), Target Shoot (-22 and .32), Memory (.29 and -.19), Number (.26, .11, and .28), and Choice Reaction (.13 and -.12)) loaded on this factor. On this factor, the reflected loadings of "time to act" were of opposite sign to the loadings for "accuracy of act." This indicates that some "slowness in responding" was correlated with "correctness of response" and/or some "quickness in responding" was correlated with "error in response."

F-4.1.5 Vocabulary Knowledge

This factor had high loadings for the ASVAB subtests Verbal (.76), Word Knowledge (.66), Paragraph Comprehension (.49), and (to a lesser degree) General Science (.35). It is suspected that the ASVAB "Verbal" is not a subtest in its own right, but some linear combination of the other three ASVAB tests. This would also explain why ASVAB Verbal had a communality of 1.04 (creating a "Heywood" case). Thus, this factor could either be one that represents the necessary correlation of the "Verbal" test with its "components, or it may also represent vocabulary knowledge above and beyond that needed for normal reading comprehension. Only three other tests loaded above .10 on this factor, and none of these loadings exceeded .138.

F-4.1.6 Technical Knowledge

Only the four ASVAB "technical knowledge" tests Autoshop Information (.34), (Mechanical Comprehension (.33), Electronic Information (.23), and General Science (.13)) loaded significantly on this factor. This factor may well represent particular technical knowledge and information which has been previously learned and stored into some soldiers' long-term memory. It appears to represent technical "book" knowledge that begets high scores on these ASVAB tests , but does not translate into improved scores on any of the other tests.

F-4.1.7 Logical Reasoning Speed

All of the tests loading significantly on this factor do require capabilities which, traditionally, have been identified as "Logical Reasoning." The test loading on this factor were ASVAB Mathematical Knowledge (.41), ASVAB Arithmetic Reasoning (.33), CORR Reasoning (.29), CORR Orientation (.27), CORR Map Test (.26), CORR Assembling Objects (.26), Memory Decision Time (.21), PSA Decision Time (.18), and ASVAB Mechanical Comprehension (.14).

F-4.1.8 Spatial/Figural Orientation

The fact that all of the CORR tests load positively (Maze Test (.51), Assembling Objects (.47), Object Rotation (.43), Reasoning (.37), Orientation (.29), and Map (.28)) on this factor strongly suggested that this factor represented the internal process required to accomplish spatial/figural orientation tasks. Several of the other tests (e.g., Target ID: Decision Time (.34) and Hit Rate (.14), PSA Decision Time (.15), Cannon Shoot (.15), and ASVAB Mechanical Comprehension (.14)) having significant loadings on this factor appear to contain test items with elements that would also require capability in "Spatial/Figural Orientation."

F-4.1.9 Eye-Movement Speed

The two test loading highest on this factor were the ASVAB Coding Speed (.53) and ASVAB Numerical Operations (.48) subtests. Enhanced performance on both of these tests would result from an ability to rapidly move the eyes back and forth between two columns of data to be compared. The only other tests loading significantly on this factor were the CORR Maze Test (.13) and PSA Trimmed Decision Time (.11). Performance on it would also be enhanced by rapid eye movements through the mazes. For these reasons, this factor is identified as "Eye Movement Speed."

F-4.1.10 Eye-Hand Coordination

The test criteria loading highest on this factor included Target Tracking (.84 and .75), Target Shoot (.44 and .35), Cannon Shoot (.32), Choice Reaction Time (.27), and Target ID Decision Time (.16). It is likely that the ability represented by this factor is one needed to visually observe a moving target and, at the "precise" time, make an appropriate manual response. While the title given to this factor is "Eye-Hand Coordination," the accurate "timing" of the manual response may well involve the capability for accurate visual extrapolation of a target's motion and then to choose to respond at the appropriate time.

F-4.1.11 Figural Recognition Time/Accuracy Tradeoff Strategy

This factor loaded only on only four of the computerized test parameters (i.e., Target Shoot: (.37 and -.34) and PSA: (.26 and -.24)). Both Target Shoot and PSA require recognition of figures. Because of the signs of the loadings of the two criterion scores for each test, it appears that this factor may represent a time versus accuracy tradeoff strategy for recognizing figures.

F-4.1.12 Decision Making Speed

Most of the parameters that purportedly measured the time to make a decision on various computerized tests loaded significantly on this factor. Those parameters included Average Trimmed Choice Reaction Time (.53), Average Trimmed Decision Time for the Memory Test (.52), Average Final Response Time for the Number Test (.24), Average Trimmed Decision Time for the PSA test (.17), and Average Trimmed Decision Time for Target ID (.14)). From these results it is likely that stable individual differences do exist in the time required to make simple decisions.

F-4.1.13 Reaction Speed

The two highest loadings for this factor were average trimmed response times for Simple (.65) and Choice (.52) Reaction tests.

F-4.1.14 Reaction Accuracy

The two highest loadings on this factor were average hit rates for Simple (.41) and Choice (.48) Reaction tests. The fact that separate factors appeared for Reaction Speed and Reaction Accuracy may represent different strategies in how these types of situations are typically handled by individuals. Reaction Speed may be physiologically determined, (i.e., some people simply react more slowly than others). Reaction Accuracy may represent a willingness to guess what the appropriate response will be, and may at times react so fast that they make an error in responding. The two factors could have been rotated to positions that would have shown one factor as an overall "goodness" (i.e., short time and high accuracy) to reaction tasks, and the remaining factor as a time/accuracy tradeoff strategy.

F-4.1.15 Hand Reach Speed

The three highest loadings on this factor were the reflected loadings for Pooled Average Movement Time (.40), Average Final Response Time for the Number Test (.31), and Average Trimmed Decision Time for the Memory Test (.27). This strongly suggests that persons scoring high on this factor would be fast in moving their hand to the response button(s) used in the computerized tests. The slight positive reflected loadings for the two Tracking criteria (.13 and .12) may indicate that hand movement speed contributes to better performance in that type of task as well.

F-4.1.16 Overall Body Size

The fact that only Height (.70) and Weight (.73) load on this factor indicates that these two parameters of overall body size are obviously related. Persons scoring high on this factor will be taller and heavier than those scoring low on it. If body strength is also related to body size, inere is little indication that any of the 37 tests required body strength.

F-4.1.17 Criteria Similarity for the "Number Test"

This factor probably does not represent anything other than the similarity between Pooled Average Operation Time (.61) and Average Initial Input Time (.54) for the Number Test. As such, it is not considered to be a real factor or a separate internal process.

F-4.1.18 Number Test Accuracy

Of all the final rotated factors, this one had the lowest eigenvalue and cannot be interpreted. The only tests having a loading on this factor greater than .14 was the criterion score for Average Hit Rate for the Number Test (.31) and ASVAB CODING Speed (.21). It may be recalled that Number Average Hit Rate was the variable that had showed unexplained loadings on factors 1 and 3, and it was the only variable having virtually no variance.

F-2. REFERENCES

Kaiser, H. F. (1959). Computer program for varimax rotation in factor analysis. Educational and Psychological Measurement, 19, 413-420.

Wing, H. (1985). Expanding the measurement of predictor space for military enlisted jobs. Symposium presented at the Annual Meeting of the American Psychological Association, Los Angles.

Wherry, Jr. R. J. (1986). Theoretical development for identifying underlying internal processes, Volume 2. Modifications to hierarchical factor analysis: Positive manifold (POSMAN) rotations.